Graph and String Parameters: Connections Between Pathwidth, Cutwidth and the Locality Number

Katrin Casel[1], Joel D. Day[2], Pamela Fleischmann[3], Tomasz Kociumaka[4], Florin Manea[5], and Markus L. Schmid[1]

[1]Humboldt-Universität zu Berlin, Berlin, Germany, Katrin.Casel@hu-berlin.de, MLSchmid@MLSchmid.de

[2]Department of Computer Science, Loughborough University, Loughborough, United Kingdom, J.Day@lboro.ac.uk

[3]Department of Computer Science, Kiel University, Kiel, Germany, fpa@informatik.uni-kiel.de

[4]Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany, tomasz.kociumaka@mpi-inf.mpg.de

[5]Computer Science Department, Universität Göttingen, Göttingen, Germany, florin.manea@informatik.uni-goettingen.de

Abstract

We investigate the locality number, a recently introduced structural parameter for strings (with applications in pattern matching with variables), and its connection to two important graph-parameters, cutwidth and pathwidth. These connections allow us to show that computing the locality number is NP-hard, but fixed-parameter tractable, if parameterised by the locality number or by the alphabet size, which has been formulated as open problems in the literature. Moreover, the locality number can be approximated with ratio $O(\sqrt{\log(\mathsf{opt})}\log(n))$.

An important aspect of our work – that is relevant in its own right and of independent interest – is that we identify connections between the string parameter of the locality number on the one hand, and the famous graph parameters of cutwidth and pathwidth, on the other hand. These two parameters have been jointly investigated in the literature and are arguably among the most central graph parameters that are based on "linearisations" of graphs. In this way, we also identify a direct approximation preserving reduction from cutwidth to pathwidth, which shows that any polynomial $f(\mathsf{opt}, |V|)$-approximation algorithm for pathwidth yields a polynomial $2f(2\,\mathsf{opt}, h)$-approximation algorithm for cutwidth on multigraphs (where h is the number of edges). In particular, this translates known approximation ratios for pathwidth into new approximation ratios for cutwidth, namely $O(\sqrt{\log(\mathsf{opt})}\log(h))$ and $O(\sqrt{\log(\mathsf{opt})}\,\mathsf{opt})$ for (multi) graphs with h edges.

1 Introduction

Graphs, on the one hand, and strings (we also use the term *word*), on the other, are two different types of data objects and they have certain particularities. Graphs seem to be more popular in fields like classical and parameterised algorithms and complexity (due to the fact that many natural graph problems are intractable), while fields like formal languages, pattern matching, verification or compression are more concerned with strings. Moreover, both the field of graph algorithms as well as string algorithms are well established and provide rich toolboxes of algorithmic techniques, but they differ in that the former is tailored to computationally hard problems (e.g., the approach of treewidth and related parameters), while the latter focuses on providing efficient data-structures for near-linear-time algorithms. Nevertheless, it is sometimes possible to bridge this divide, i.e., by "flattening" a graph into a sequential form, or by "inflating" a string into a graph, to make use of

Marked word	Marking sequence	Marking number
adabadbdaecbcb		0
ad̄abad̄bdaecb̄cb̄	b	4
ad̄ab̄ad̄bdaecb̄cb	c	3
ad̄ab̄ad̄bdaec̄bcb	e	3
ad̄ab̄ad̄bdaec̄bcb	d	4
adabadbdaecbcb	a	1

Marked word	Marking sequence	Marking number
adabadbdaecbcb		0
ad̄abad̄bd̄aecbcb	d	3
ad̄abad̄bd̄aecbcb	a	3
ad̄abadbdaecb̄cb̄	b	3
ad̄abadbdaec̄bcb	c	2
adabadbdaecbcb	e	1

Figure 1: An illustration of the marking sequence $(\mathtt{b}, \mathtt{c}, \mathtt{e}, \mathtt{d}, \mathtt{a})$ with marking number of 4 for the word $\beta = \mathtt{adabadbdaecbcb}$ (left side), and the marking sequence $(\mathtt{d}, \mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{e})$ with marking number of 3 for the word β (right side).

respective algorithmic techniques otherwise not applicable. This paradigm shift may provide the necessary leverage for new algorithmic approaches.

In this paper, we are concerned with certain structural parameters (and the problems of computing them) for graphs and strings: the *cutwidth* $\mathsf{cw}(G)$ of a graph G (i.e., the maximum number of "stacked" edges if the vertices of a graph are drawn on a straight line), the *pathwidth* $\mathsf{pw}(G)$ of a graph G (i.e., the minimum width of a tree decomposition the tree structure of which is a path), and the *locality number* $\mathsf{loc}(\alpha)$ of a string α (explained in more detail in the next paragraph; formal definitions follow in Section 2). By Cutwidth, Pathwidth and Loc, we denote the corresponding decision problems (i. e., checking whether $\mathsf{cw}(G) \leq k$, $\mathsf{pw}(G) \leq k$, or $\mathsf{loc}(\alpha) \leq k$, respectively) and with the prefix Min, we refer to the minimisation variants (for which we are mainly interested in approximation algorithms). The two former graph-parameters are very classical. Pathwidth is a simple (yet still hard to compute) subvariant of treewidth, which measures how much a graph resembles a path. The problems Pathwidth and MinPathwidth are intensively studied (in terms of exact, parameterised and approximation algorithms) and have numerous applications (see the surveys and textbook [10, 37, 8]). Cutwidth is the best-known example of a whole class of so-called *graph layout problems* (see the survey [18, 43] for detailed information), which are studied since the 1970s and were originally motivated by questions of circuit layouts.

The locality number is rather new and we shall discuss it in more detail. A word is k-local if there exists an order of its symbols such that, if we *mark* the symbols in the respective order (which is called a *marking sequence*), at each stage there are at most k contiguous blocks of marked symbols in the word. This k is called the *marking number* of that marking sequence. The *locality number* of a word is the smallest k for which that word is k-local, or, in other words, the minimum marking number over all marking sequences. For example, the marking sequence $\sigma = (\mathtt{x}, \mathtt{y}, \mathtt{z})$ marks $\alpha = \mathtt{xyxyzxz}$ as follows (marked blocks are illustrated by overlines):

$$\mathtt{xyxyzxz} \rightsquigarrow \mathtt{\bar{x}y\bar{x}yz\bar{x}z} \rightsquigarrow \mathtt{\overline{xyx}yz\bar{x}z} \rightsquigarrow \mathtt{\overline{xyxyzxz}}\,;$$

thus, the marking number of σ is 3. In fact, all marking sequences for α have a marking number of 3, except $(\mathtt{y}, \mathtt{x}, \mathtt{z})$, for which it is 2:

$$\mathtt{x\bar{y}x\bar{y}zxz} \rightsquigarrow \mathtt{\overline{xyx}\bar{y}z\bar{x}z} \rightsquigarrow \mathtt{\overline{xyxyzxz}}\,.$$

Thus, the locality number of α is $\mathsf{loc}(\alpha) = 2$. For the slightly more complicated word $\beta = \mathtt{adabadbdaecbcb}$, it can be verified that $\mathsf{loc}(\beta) = 3$ (see Figure 1 for an illustration of two marking sequences for β with marking numbers 4 and 3, respectively).

The locality number has applications in pattern matching with variables [15]. A *pattern* is a word that consists of *terminal symbols* (e.g., $\mathtt{a}, \mathtt{b}, \mathtt{c}$), treated as constants, and *variables* (e.g., $x_1, x_2, x_3, \ldots$). A pattern is mapped to a word by substituting the variables by strings of terminals. For example, $x_1x_1\mathtt{bab}x_2x_2$ can be mapped to $\mathtt{acacbabcc}$ by the substitution $(x_1 \to \mathtt{ac}, x_2 \to \mathtt{c})$. Deciding whether a given pattern matches (i.e., can be mapped to) a given word is one of the most important problems that arise in the study of patterns with variables (note that the concept of patterns with variables arises in several different domains like combinatorics on words (word equations [33], unavoidable patterns [39]), pattern matching [1], language theory [2], learning theory [2, 20, 42, 46, 34, 23], database theory [7], as well as in practice, e.g., extended regular expressions with backreferences [27, 28, 48, 29], used in programming languages like Perl, Java, Python,

etc.). Unfortunately, the *matching problem* is NP-complete [2] in general (it is also NP-complete for strongly restricted variants [24, 22] and also intractable in the parameterised setting [25]); see also [41] for a survey. As demonstrated in [47], for the matching problem a paradigm shift as sketched in the first paragraph above yields a very promising algorithmic approach. More precisely, any class of patterns with bounded treewidth (for suitable graph representations) can be matched in polynomial-time. However, computing (and therefore algorithmically exploiting) the treewidth of a pattern is difficult (see the discussion in [22, 47]), which motivates more direct string-parameters that bound the treewidth and are simple to compute (virtually all known structural parameters that lead to tractability [15, 22, 47, 49] are of this kind (the efficiently matchable classes investigated in [16] are one of the rare exceptions)). This also establishes an interesting connection between ad-hoc string parameters and the more general (and much better studied) graph parameter treewidth. The locality number is a simple parameter directly defined on strings, it bounds the treewidth and the corresponding marking sequences can be seen as instructions for a dynamic programming algorithm. However, compared to other "tractability-parameters", it seems to cover best the treewidth of a string, but whether it can be efficiently computed is unclear.

In this paper, we investigate the problem of computing the locality number (in the exact sense as well as fixed-parameter algorithms and approximations) and, by doing so, we establish an interesting connection to the graph parameters cutwidth and pathwidth with algorithmic implications for approximating cutwidth. In the following, we first discuss related results in more detail and then outline our respective contributions.

Note that a conference version of this paper has been published in ICALP 2019 [13].

1.1 Known Results and Open Questions

For Loc, only exact exponential-time algorithms are known and whether it can be solved in polynomial-time, or whether it is at least fixed-parameter tractable is mentioned as open problems in [15]. Approximation algorithms have not yet been considered. Addressing these questions is the main purpose of this paper.

Pathwidth and Cutwidth are NP-complete, but fixed-parameter tractable with respect to the standard parameters $\mathsf{pw}(G)$ or $\mathsf{cw}(G)$, respectively (even with "linear" fpt-time $g(k)\,\mathrm{O}(n)$ [9, 11, 51]). With respect to approximation, their minimisation variants have received a lot of attention, mainly because they yield (like many other graph parameters) general algorithmic approaches for numerous graph problems, i.e., a good linear arrangement or path-decomposition can often be used for a dynamic programming (or even divide and conquer) algorithm. More generally speaking, pathwidth and cutwidth are related to the more fundamental concepts of small balanced vertex or edge separators for graphs (i.e., a small set of vertices (or edges, respectively) that, if removed, divides the graph into two parts of roughly the same size. More precisely, $\mathsf{pw}(G)$ and $\mathsf{cw}(G)$ are upper bounds for the smallest balanced *vertex* separator of G and the smallest balanced *edge* separator of G, respectively (see [21] for further details and explanations of the algorithmic relevance of balanced separators).

With respect to MinPathwidth, there is an approximation algorithm with ratio $\mathrm{O}(\log n\sqrt{\log \mathit{opt}})$ (see [21, Corollary 6.5]) and an approximation algorithm with ratio $\mathrm{O}(\mathsf{tw}\sqrt{\log \mathsf{tw}})$, where tw is the treewidth of the graph (see [30]).

For MinCutwidth, there is an $\mathrm{O}(\sqrt{\log(n)}\log(n))$ approximation algorithm. This follows from using the $\mathrm{O}(\sqrt{\log n})$-approximation for sparsest cut of [4] as described in the cutwidth approximation of [38, Section 3.8], which adds a $\log(n)$-factor.

1.2 Our Contributions

There are two natural approaches to represent a word α over alphabet Σ as a graph $G_\alpha = (V_\alpha, E_\alpha)$. The first option is to represent α's positions as vertices, i. e., $V_\alpha = \{1, 2, \ldots, |\alpha|\}$, and then somehow use the edges to represent the actual symbols on these position. We present such a reduction to relate the locality number of words with the pathwidth of graphs. More precisely, we transform a word α into a graph such that $|E_\alpha| = \mathrm{O}(|\alpha|^2)$ and $\mathsf{loc}(\alpha) \leq \mathsf{pw}(G_\alpha) \leq 2\,\mathsf{loc}(\alpha)$.

The second option is to use a vertex per symbol that occurs in α, i. e., $V_\alpha = \Sigma$, and somehow use the edges to encode where these symbols occur in the word. By such a reduction with $|E_\alpha| =$

$O(|\alpha|)$, we can relate the locality number of words with the cutwidth of graphs in the sense that $\mathsf{cw}(G_\alpha) = 2\,\mathsf{loc}(\alpha)$.

Since these reductions are parameterised reductions and also approximation preserving, known upper bounds for the problems of computing or approximating the pathwidth or cutwidth of graphs carry over to the problem of computing or approximating the locality number of words. More precisely, we can conclude that Loc is fixed-parameter tractable if parameterised by $|\Sigma|$ or by the locality number (answering the respective open problem from [15]), and also that there is a polynomial-time $O(\sqrt{\log(\mathsf{opt})}\log(n))$-approximation algorithm for MinLoc.

In addition to these reductions, we can also show how an arbitrary multi-graph $G = (V, E)$ and an edge $e \in E$ can be represented by a word $\alpha_{G,e}$ over alphabet V, of length $|E|$ and with $\mathsf{cw}(G) \leq \mathsf{loc}(\alpha_{G,e}) \leq \mathsf{cw}(G) + 1$. Moreover, there must be an edge $e \in E$ such that $\mathsf{loc}(\alpha_{G,e}) = \mathsf{cw}(G)$. This describes an (approximation preserving) Turing-reduction from Cutwidth to Loc which allows us to conclude that Loc is NP-complete (which solves the other open problem from [15]).

Even though the reduction from MinLoc to MinPathwidth yields an $O(\sqrt{\log(\mathsf{opt})}\log(n))$-approximation algorithm for MinLoc, it is also important to directly investigate whether obvious greedy strategies for constructing marking sequences (e. g., always marking a symbol next that leads to the smallest number of marked blocks) yield good approximation ratios. On the one hand, if such strategies fail, we can rule them out as possible approximation algorithms for computing the locality number, and, on the other hand, if such simple strategies work, then, due to the reduction from MinCutwidth to MinLoc, this might open a new angle to the approximation of cutwidth. Unfortunately, we can formally show that many natural candidates for greedy strategies fail to yield promising approximation algorithms (and are therefore also not helpful for cutwidth approximation).

Expecting an improvement of cutwidth approximation – a heavily researched area – by translating the problem into a string problem and then investigating the approximability of this string problem seems naive. This makes it even more surprising that linking cutwidth with pathwidth via the locality number is in fact helpful for cutwidth approximation. More precisely, by plugging together our reductions from MinCutwidth to MinLoc and from MinLoc to MinPathwidth, we obtain a reduction which directly transfers approximation results from MinPathwidth (e. g., the ones of [21, 30]; see the discussion of Section 1.1) to MinCutwidth. On the one hand, this reduction yields new concrete approximation ratios for cutwidth (mentioned in more detail below), but, on the other hand, it also shows that any future improvement of pathwidth approximation directly carries over to cutwidth approximation (although there is a constant factor involved for constant factor approximations of pathwidth).[1]

A reason why this direct reduction from cutwidth to pathwidth has been overlooked might be that the literature on cutwidth and pathwidth approximation is focussed on more general approximation techniques (i. e., vertex and edge separators), which then yield approximation algorithms for these graph parameters. Another reason might be that this relation is less obvious on the graph level and becomes more apparent if linked via the string parameter of locality, as in our considerations. Nevertheless, since pathwidth and cutwidth are such crucial parameters for graph algorithms, we also translate our locality based reduction into one from graphs to graphs directly.

We conclude this subsection by summarising the main results of this work:

- We present approximation preserving reductions from Loc to Cutwidth and Pathwidth, and an approximation preserving reduction from Cutwidth to Loc.
- Loc is NP-complete, but fixed-parameter tractable if parameterised by $|\Sigma|$ or by the locality number.
- There is a polynomial-time $O(\sqrt{\log(\mathsf{opt})}\log(n))$-approximation algorithm for MinLoc.
- Many obvious greedy strategies for MinLoc do not yield good approximation algorithms.
- We present an approximation preserving reduction from Cutwidth to Pathwidth.
- There is a polynomial-time $O(\sqrt{\log(\mathsf{opt})}\log(n))$-approximation algorithm and a polynomial-time $O(\sqrt{\log(\mathsf{opt})}\,\mathsf{opt})$-approximation algorithm for MinCutwidth.

[1] Note that both the pathwidth and the cutwidth is NP-hard to approximate to within a constant factor [52].

1.3 Organisation of the Paper

In Section 2, we give basic definitions (including the central parameters of the locality number, the cutwidth and the pathwidth). In the next Section 3, we discuss the concept of the locality number with some examples and some word combinatorial considerations. The purpose of this section is to develop a better understanding of this parameter for readers less familiar with string parameters and combinatorics on words (the technical statements of this section are formally proven in the appendix).

The main results are presented in Sections 4, 5 and 6. First, in Section 4, we present the reductions from Loc to Cutwidth and vice versa, and we discuss the consequences of these reductions. Then, in Section 5, we show how Loc can be reduced to Pathwidth, which yields an approximation algorithm for computing the locality number; furthermore, we investigate the performance of direct greedy strategies for approximating the locality number. Finally, since we consider this of high importance independent of the locality number, we provide a direct reduction from cutwidth to pathwidth in Section 6.

In Section 7, we conclude the paper by discussing some related topics and possible further research questions.

2 Preliminaries

We now define basic concepts of complexity theory, some basics about string algorithms and the locality number, the central graph parameters cutwidth and pathwidth, and the formal definitions of the decision and minimisation problems to be investigated.

2.1 (Parameterised) Complexity Theory and Approximation Algorithms

We briefly summarise the fundamentals of parameterised complexity [26, 19] and approximation algorithms [5].

A *parameterised problem* is a decision problem with instances (x, k), where x is the actual input and $k \in \mathbb{N}$ is the *parameter*. A parameterised problem P is *fixed-parameter tractable* if there is an *fpt-algorithm* for it, i.e., one that solves P on input (x, k) in time $f(k) \cdot p(|x|)$ for a recursive function f and a polynomial p. In this case, we also say that the parameterised problem P can be solved with *fpt-running-time* $f(k) \cdot p(|x|)$.

We use the $\mathrm{O}^*(\cdot)$ notation which hides multiplicative factors polynomial in $|x|$.

A minimisation problem P is a triple (I, S, m), where I is the *set of instances*, S is a function that maps instances $x \in I$ to the *set of feasible solutions* for x, and m is the *objective function* that maps pairs (x, y) with $x \in I$ and $y \in S(x)$ to a positive rational number. For every $x \in I$, we denote $m^*(x) = \min\{m(x, y) \colon y \in S(x)\}$. For $x \in I$ and $y \in S(x)$, the value $R(x, y) = \frac{m(x,y)}{m^*(x)}$ is the *performance ratio* of y with respect to x. An algorithm $\mathcal{A}$ is an *approximation algorithm* for P with ratio $r : \mathbb{N} \to \mathbb{Q}$ (or an r-approximation algorithm, for short) if, for every $x \in I$, $\mathcal{A}(x) = y \in S(x)$, and $R(x, y) \leq r(|x|)$. We also let r be of the form $\mathbb{Q} \times \mathbb{N} \to \mathbb{Q}$ when the ratio r depends on $m^*(x)$ and $|x|$; in this case, we write $r(\mathsf{opt}, |x|)$. We further assume that the function r is monotonically non-decreasing. Unless stated otherwise, all approximation algorithms run in polynomial time with respect to $|x|$.

2.2 Basic String Definitions and Locality

The set of strings (or words) over an alphabet X is denoted by X^*, by $|\alpha|$ we denote the length of a word α, $\mathsf{alph}(\alpha)$ is the smallest alphabet X with $\alpha \in X^*$, and ε denotes the empty word with $|\varepsilon| = 0$. A string β is called a *factor* of α if $\alpha = \alpha'\beta\alpha''$; if $\alpha' = \varepsilon$ or $\alpha'' = \varepsilon$, then β is a *prefix* or a *suffix* of α, respectively. For a position j, $1 \leq j \leq |\alpha|$, we refer to the symbol at position j of α by the expression $\alpha[j]$, and $\alpha[j..j'] = \alpha[j]\alpha[j+1]\dots\alpha[j']$, $1 \leq j \leq j' \leq |\alpha|$. For a word α and $x \in \mathsf{alph}(\alpha)$, let $\mathsf{ps}_x(\alpha) = \{i \mid 1 \leq i \leq |\alpha|, \alpha[i] = x\}$ be the set of all positions where x occurs in α. For a word α, let $\alpha^0 = \varepsilon$ and $\alpha^{i+1} = \alpha\alpha^i$ for $i \geq 0$.

Let α be a word and let $X = \mathsf{alph}(\alpha) = \{x_1, x_2, \dots, x_n\}$. A *marking sequence* (*over* X) of α is an enumeration, or ordering on the letters from X, and hence may be represented either as an

ordered list of the letters or, equivalently, as a bijection $\sigma : \{1, 2, \ldots, |X|\} \to \{1, 2, \ldots, |X|\}$. For a marking sequence $\sigma = (x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(m)})$, a word α and every i with $1 \leq i \leq m$, by *stage* i *of* σ we denote the word α with exactly positions $\bigcup_{j=1}^{i} \mathsf{ps}_{x_{\sigma(j)}}(\alpha)$ marked. The *marking number* $\pi_\sigma(\alpha)$ (of σ with respect to α) is the maximum number of marked blocks in any stage i of σ. We say that α is k-local if and only if, for some marking sequence σ, we have $\pi_\sigma(\alpha) \leq k$, and the smallest k such that α is k-local is the *locality number* of α, denoted by $\mathsf{loc}(\alpha)$. We say that a word w is *strictly* k-local, if $\mathsf{loc}(\alpha) = k$. A marking sequence σ with $\pi_\sigma(\alpha) = \mathsf{loc}(\alpha)$ is *optimal* (for α). For illustration, see also the examples given in Section 1.

For a word α, the *condensed form of* α, denoted by $\mathsf{cond}(\alpha)$, is obtained by replacing every maximal factor x^k with $x \in \mathsf{alph}(\alpha)$ by x. For example, $\mathsf{cond}(x_1x_1x_2x_2x_2x_1x_2x_2) = x_1x_2x_1x_2$. A word α is *condensed* if $\alpha = \mathsf{cond}(\alpha)$.

Observation 2.1. For a word $\alpha \in X^*$ and any marking sequence σ over X, we have $\pi_\sigma(\mathsf{cond}(\alpha)) = \pi_\sigma(\alpha)$. Moreover, if α is condensed, then the maximum number of occurrences of any symbol in α is bounded by $2\,\mathsf{loc}(\alpha)$ (see [15] for details). In particular, this means that for condensed words $\alpha \in X^*$, we have that $|\alpha| = \mathrm{O}(|X|\,\mathsf{loc}(\alpha))$.

Observation 2.1 justifies that in the following, we are only concerned with condensed words (and therefore words with at most $2\,\mathsf{loc}(\alpha)$ occurrences per symbol and total length of at most $|X|2\,\mathsf{loc}(\alpha)$). In particular, for any word α we can compute $\mathsf{cond}(\alpha)$ in time $\mathrm{O}(|\alpha|)$; thus algorithms for computing the locality number (and the respective marking sequences) for *condensed* words extend to algorithms for general words. For the sake of convenience, in the following we shall only use the term *word* and keep in mind that we always talk about condensed words.

2.3 Basic Graph Definitions and Graph Parameters

Let $G = (V, E)$ be a (multi)graph with the vertices $V = \{v_1, \ldots, v_n\}$. A *cut* of G is a partition (V_1, V_2) of V into two disjoint subsets V_1, V_2, $V_1 \cup V_2 = V$; the (multi)set of edges $\mathcal{C}(V_1, V_2) = \{\{x, y\} \in E \mid x \in V_1, y \in V_2\}$ is called the *cut-set* or the *(multi)set of edges crossing the cut*, while V_1 and V_2 are called the *sides* of the cut. The *size* of this cut is the number of crossing edges, i.e., $|\,\mathcal{C}(V_1, V_2)|$. A *linear arrangement* of the (multi)graph G is a sequence $(v_{j_1}, v_{j_2}, \ldots, v_{j_n})$, where $(j_1, j_2, \ldots, j_n)$ is a permutation of $(1, 2, \ldots, n)$. For a linear arrangement $L = (v_{j_1}, v_{j_2}, \ldots, v_{j_n})$, let $L(i) = \{v_{j_1}, v_{j_2}, \ldots, v_{j_i}\}$. For every i, $1 \leq i < n$, we consider the cut $(L(i), V \setminus L(i))$ of G, and denote the cut-set $\mathcal{C}_L(i) = \mathcal{C}(L(i), V \setminus L(i))$ (for technical reasons, we also set $\mathcal{C}_L(0) = \mathcal{C}_L(n) = \emptyset$). We define the *cutwidth* of L by $\mathsf{cw}(L) = \max\{|\,\mathcal{C}_L(i)| \mid 0 \leq i \leq n\}$. Finally, the cutwidth of G is the minimum over all cutwidths of linear arrangements of G, i.e., $\mathsf{cw}(G) = \min\{\mathsf{cw}(L) \mid L \text{ is a linear arrangement for } G\}$.

Let us discuss an example. To this end, let $H = (V, E)$ with $V = \{u, v, w, x, y, z\}$ and the edges E are as illustrated in Figure 2. A possible linear arrangement for H is $L = (u, v, w, x, y, z)$ with $|\,\mathcal{C}_L(1)| = 3$, $|\,\mathcal{C}_L(2)| = 5$, $|\,\mathcal{C}_L(3)| = 5$, $|\,\mathcal{C}_L(4)| = 2$ and $|\,\mathcal{C}_L(5)| = 2$; thus, $\mathsf{cw}(L) = 5$ (a cut with maximum size is $(L(3), V \setminus L(3))$, as illustrated by a vertical line in Figure 2). Another linear arrangement is $L' = (w, u, x, v, y, z)$ with $\mathsf{cw}(L') = 3$ (see Figure 2). Moreover, it can be verified that $\mathsf{cw}(H) = 3$.

A path decomposition (see [11]) of a connected graph $G = (V, E)$ is a tree decomposition whose underlying tree is a path, i.e., a sequence $Q = (B_0, B_1, \ldots, B_m)$ (of *bags*) with $B_i \subseteq V$, $0 \leq i \leq m$, satisfying the following two properties:

- *Cover property*: for every $\{u, v\} \in E$, there is an index i, $0 \leq i \leq m$, with $\{u, v\} \subseteq B_i$.
- *Connectivity property*: for every $v \in V$, there exist indices i_v and j_v, $0 \leq i_v \leq j_v \leq m$, such that $\{j \mid v \in B_j\} = \{i \mid i_v \leq i \leq j_v\}$. In other words, the bags that contain v occur on consecutive positions in $(B_0, \ldots, B_m)$.

The *width* of a path decomposition Q is $\mathsf{w}(Q) = \max\{|B_i| \mid 0 \leq i \leq m\} - 1$, and the *pathwidth* of a graph G is $\mathsf{pw}(G) = \min\{\mathsf{w}(Q) \mid Q \text{ is a path decomposition of } G\}$. A path decomposition is *nice* if $B_0 = B_m = \emptyset$ and, for every i, $1 \leq i \leq m$, either $B_i = B_{i-1} \cup \{v\}$ or $B_i = B_{i-1} \setminus \{v\}$, for some $v \in V$. We further use $|Q| = \sum_{i=1}^{n} |B_i|$.

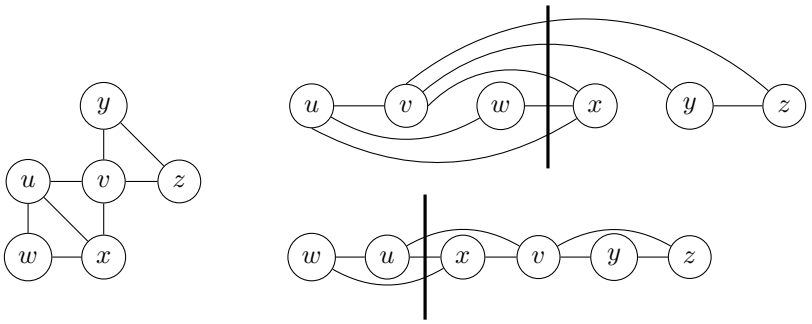

Figure 2: A graph H and two possible linear arrangements with cuts of maximum size illustrated by vertical lines.

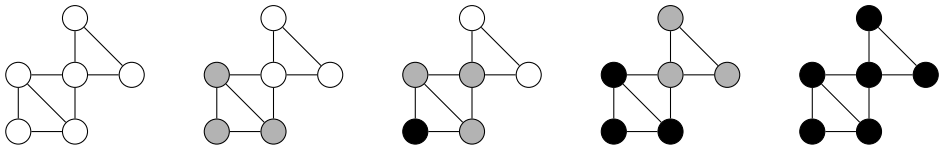

Figure 3: The path decomposition $(\{u, w, x\}, \{u, v, x\}, \{v, y, z\})$ for graph H (see Figure 2) as a pd-marking scheme. White vertices are `open`, grey vertices are `active`, and black vertices are `closed`. In order to see that this is a pd-marking scheme, it is sufficient to observe that for every edge there is a step in the pd-marking scheme where both incident vertices are grey (i. e., `active`).

For example, $(\{u, w, x\}, \{u, v, x\}, \{v, y, z\})$ is a width-2 path decomposition for the graph H defined above (see also Figure 2); as can be easily seen, $\mathsf{pw}(H) = 2$.

In this paper, it shall be convenient to interpret path decompositions as *marking schemes* of V in the following way. Every vertex from V can be marked as `open`, as `active` or as `closed`. Initially, every vertex is `open`. Only `open` vertices can be set to `active`, only `active` vertices can be set to `closed`, and in the end of the marking scheme, all vertices must be `closed`. In each step of the marking scheme, we allow an arbitrary number of vertices to be set from `open` to `active`, and an arbitrary number of vertices to be set from `active` to `closed`. Any such marking scheme translates into a sequence $Q = (B_0, B_1, \ldots, B_m)$ with $B_i \subseteq V$, $0 \leq i \leq m$, by letting B_i contain exactly the `active` vertices of step i of the marking scheme. Obviously, Q satisfies the connectivity property (this is a direct consequence from the fact that every vertex is marked `active` at some point and as soon as it is marked `closed`, it is never marked as `active` again). If Q also satisfies the cover property, then Q is a path decomposition, and in this case we call the corresponding marking scheme a *pd-marking scheme*. The width of a pd-marking scheme is then the maximum number of vertices which are marked `active` at the same time minus one.

For example, the path decomposition $(\{u, w, x\}, \{u, v, x\}, \{v, y, z\})$ for graph H can be represented as a pd-marking scheme as illustrated in Figure 3 (for convenience, we omit the vertex labels; see also Figure 2 for an illustration of H).

Both the locality number of a word and the pathwidth of a graph is defined via markings. In order to avoid confusion, we therefore use different terminology to distinguish between these two concepts (see also the terminology defined in Section 2.2): The markings for words are called *marking sequences*, while the markings for graphs are called *pd-marking schemes*; the versions of a word during a marking sequence are called the *stages* (of the marking sequence), while the different marked version of a graph during a pd-marking scheme are called the *steps* (of the pd-marking scheme).

2.4 Problem Definitions

We next formally define the computational problems of computing the parameters defined above. By Loc, Cutwidth and Pathwidth, we denote the problems to check for a given word α or graph G and integer $k \in \mathbb{N}$, whether $\mathsf{loc}(\alpha) \leq k$, $\mathsf{cw}(G) \leq k$, and $\mathsf{pw}(G) \leq k$, respectively. Note that since we can assume that $k \leq |\alpha|$ and $k \leq |G|$, whether k is given in binary or unary has no impact on the complexity. With the prefix Min, we refer to the minimisation variants. More precisely, MinLoc $= (I, S, m)$, where I is the set of words, $S(\alpha)$ is the set of all marking sequences for α and $m(\alpha, \sigma) = \pi_\sigma(\alpha)$ (note that $m^*(\alpha) = \mathsf{loc}(\alpha)$); MinCutwidth $= (I, S, m)$, where I are all multigraphs, $S(G)$ is the set of linear arrangements of G, and $m(G, L) = \mathsf{cw}(L)$ (note that $m^*(G) = \mathsf{cw}(G)$); finally, MinPathwidth $= (I, S, m)$, where I are all graphs, $S(G)$ is the set of path decompositions of G, and $m(G, Q) = \mathsf{w}(Q)$ (note that $m^*(G) = \mathsf{pw}(G)$).

3 Examples and Word Combinatorial Considerations

In this section, we discuss some examples that illustrate the concepts of marking sequences and the locality number, and we also discuss some word combinatorial properties related to the locality number. Note that for illustration purposes, the example words considered in this section are not necessarily condensed.

It is easy to see that 1-locality implies some sort of *palindromic* structure of a word. For example, palindromes like the English words *radar*, *refer* and *rotator* are obviously 1-local, while the palindrome `abababa` is obviously not 1-local. Moreover, also 1-local non-palindromes, like the word *blender*, have some palindromic structure. More precisely, it can be shown that a word w is 1-local if and only if $w \in \{a_1\}^*\{a_2\}^* \dots \{a_n\}^*\{a_{n-1}\}^* \dots \{a_1\}^*$, such that $\{a_1, a_2, \dots, a_i\} \cap \{a_{i+1}, a_{i+2}, \dots, a_n\} = \emptyset$ for every $1 \leq i \leq n$. An alternative equivalent point of view is that 1-local words are necessarily of the form $y^\ell \alpha y^r$, where α is 1-local with $y \notin \mathsf{alph}(\alpha)$. For further details, we refer to [15], where the structure of 1-local and 2-local words is characterised.

Determining structural properties that lead to high locality is more challenging. The Finnish word *tutustuttu* (perfect passive of *tutustua*—to meet) is 4-local, while

pneumonoultramicroscopicsilicovolcanoconiosis

is an (English) 8-local word, and

lentokonesuihkuturbiinimoottoriapumekaanikkoaliupseerioppilas

is a 10-local (Finnish) word. In general, in order to have a high locality number, a word needs to contain many alternating occurrences of (at least) two letters. For instance, $(x_1x_2)^n$ is n-local. However, the number of occurrences of a letter alone is not always a good indicator of the locality of a word. The German word *Einzelelement* (a basic component of a construction) has 5 occurrences of e, but is only 3-local, as witnessed by marking sequence (l,m,e,i,n,z,t):

$$\texttt{Einzelelement} \rightsquigarrow \texttt{Einzel}\overline{\texttt{e}}\texttt{l}\overline{\texttt{e}}\texttt{ment} \rightsquigarrow \texttt{Einzel}\overline{\texttt{e}}\texttt{l}\overline{\texttt{e}}\texttt{m}\overline{\texttt{e}}\texttt{nt} \rightsquigarrow \overline{\texttt{Einzelelement}} \rightsquigarrow$$
$$\overline{\texttt{E}}\texttt{inz}\overline{\texttt{elelement}} \rightsquigarrow \overline{\texttt{Ein}}\texttt{z}\overline{\texttt{elelement}} \rightsquigarrow \overline{\texttt{Einzelelement}} \rightsquigarrow \overline{\texttt{Einzelelement}}$$

For this example marking sequence, it is worth noting that marking the many occurrences of e joins several individual marked blocks into one marked block. This also intuitively explains the correspondence between the locality number and the maximum number of occurrences per symbol (in condensed words): if there are $2k$ occurrences of a symbol, then, by marking this symbol either at least k new marked blocks are created, or at least k marked blocks must already exist before marking this symbol (see Observation 2.1).

A repetitive structure often leads to high locality. For example, note that *tutustuttu* from above is nearly a repetition. Regarding the question of how repetitions of a word affect its locality number, we can show the following result (see the Appendix for a proof).

Lemma 3.1. *Let $w = u^i$ be the i-times repetition of $u \in X^*$ and $i \in \mathbb{N}$. If u is strictly k-local then $\mathsf{loc}(w) \in \{ik - i + 1, ik\}$.*

The well-known *Zimin words* [39] also have high locality numbers compared to their lengths. These words are important in the domain of avoidability, as it was shown that a terminal-free pattern is unavoidable (i.e., it occurs in every infinite word over a large enough finite alphabet) if and only if it occurs in a Zimin word. The Zimin words Z_i, for $i \in \mathbb{N}$, are inductively defined by $Z_1 = x_1$ and $Z_{i+1} = Z_i x_{i+1} Z_i$. Clearly, $|Z_i| = 2^i - 1$ for all $i \in \mathbb{N}$. Regarding the locality of Z_i, note that marking x_2 leads to 2^{i-2} marked blocks; further, marking x_1 first and then the remaining symbols in an arbitrary order only extends or joins marked blocks. Thus, we obtain a sequence with marking number 2^{i-2}. In fact, with respect to the locality of Zimin words, we can show the following result (see the Appendix for a proof).

Lemma 3.2. $\mathsf{loc}(Z_i) = \frac{|Z_i|+1}{4} = 2^{i-2}$ *for* $i \in \mathbb{N}_{\geq 2}$.

Notice that both Zimin words and 1-local words have an obvious palindromic structure. However, in the Zimin words, the letters occur multiple times, but not in large blocks, while in 1-local words there are at most 2 blocks of each letter. With respect to palindromes, we can show the following general result (see the Appendix for a proof).

Lemma 3.3. *If w is a palindrome, with $w = uau^R$ or $w = uu^R$ (u^R denotes the reversal of u), and $\mathsf{loc}(u) = k$, then $\mathsf{loc}(w) \in \{2k-1, 2k, 2k+1\}$.*

4 Locality and Cutwidth

In this section, we introduce polynomial-time reductions from the problem of computing the locality number of a word to the problem of computing the cutwidth of a graph, and vice versa. This establishes a close relationship between these two problems (and their corresponding parameters), which lets us derive several upper and lower complexity bounds for Loc. We also discuss the approximation-preserving properties of our reductions, which shall be important later on.

4.1 Reducing Locality Number to Cutwidth

First, we show a reduction from Loc to Cutwidth. For a word α and an integer $k \in \mathbb{N}$, we build a multigraph $H_{\alpha,k} = (V, E)$ whose set of nodes $V = \mathsf{alph}(\alpha) \cup \{\$, \#\}$ consists of symbols occurring in α and two additional characters $\$, \# \notin \mathsf{alph}(\alpha)$. The multiset of edges E is constructed as follows. We scan through the word from left to right, and for every individual occurrence of a size-2 factor xy (often called *digrams* in the literature on strings), we add an edge between vertices x and y. Since the edges are undirected, this means that both an occurrence of xy and an occurrence of yx will cause the addition of and edge $\{x, y\}$. Moreover, independent of α's structure, we add $2k$ edges between \$ and #, one edge between \$ and the first letter of α, and one edge between \$ and the last letter of α.

Let us clarify this reduction with an example. Let $\alpha = \mathtt{abcbcdbada}$ and $k = 2$. This means that $H_{\alpha,k} = (V, E)$ with $V = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \$, \#\}$. There are $2k = 4$ edges between vertices \$ and #. Moreover, two edges connect \$ with the first and last letter of α, respectively, which, in this case, is both the letter a, which means there are two edges between \$ and a. The edges that actually depend on α's structure are obtained by scanning through α from left to right with a window of size 2, and adding the respective edge for each size-2 factor that we see in this window. For our example, we thus add an edge between a and b due to factor ab, then an edge between b and c due to the factor bc, then another edge between b and c due to the factor cb, and so on. This results in the multigraph shown on the left of Figure 4.

We claim that $\mathsf{loc}(\alpha) \leq k$ if and only if the smallest cutwidth of any linear arrangement of the graph $H_{\alpha,k}$ is exactly $2k$. Before formally proving this, let us discuss this on an intuitive level with our example. It is not hard to see that any linear arrangement of $H_{\alpha,k}$ with cutwidth $2k$ must start with vertices $\#, \$$ or end with vertices $\$, \#$ (this is due to the edges between # and \$). This means that any linear arrangement of $H_{\alpha,k}$ with cutwidth $2k$ induces a marking sequence for α. The optimal linear arrangement of $H_{\alpha,k}$ with cutwidth 4 shown on the right side of Figure 4 thus induces the marking sequence (c, b, d, a). If we now carry out this marking sequence on α and in parallel move through $H_{\alpha,k}$'s linear arrangement from left to right, we can see that the vertices to the left of our current position correspond to marked symbols, the vertices to the right

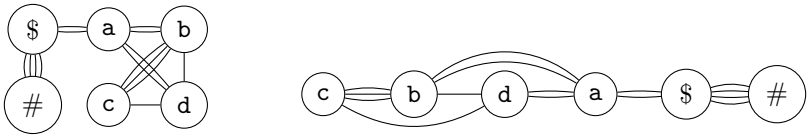

Figure 4: The graph $H_{\alpha,k}$ for $\alpha = \texttt{abcbcdbada}$ and $k = 2$; an optimal linear arrangement of $H_{\alpha,k}$ with cutwidth 4 induces the optimal marking sequence $(\texttt{c}, \texttt{b}, \texttt{d}, \texttt{a})$ for α with marking number 2.

of our current position correspond to unmarked symbols, and there is exactly one crossing edge per boundary between a marked and an unmarked block in α (in particular, all non-crossing edges to the left and all non-crossing edges to the right correspond to size-2 factors that are completely contained in a marked block or an unmarked block, respectively). This means that the cuts of the linear arrangement have size twice the current number of marked blocks in the marked version of α; thus, the size of any cut is at most $2k$, where k is the marking number of the corresponding marking sequence. Note, however, that we need the edges between \$ and the first and last symbol to also enforce two crossing edges per marked prefix or suffix, and the $2k$ edges between \$ and # force the cutwidth to be at least $2k$. In particular, for our example, the linear arrangement has a cutwidth of 4, while the corresponding marking sequence has a marking number of 2 (which is optimal for α). We shall now provide a formal proof for our claim.

Lemma 4.1. *The graph $H_{\alpha,k}$ satisfies $\mathsf{cw}(H_{\alpha,k}) = 2k$ if and only if $\mathsf{loc}(\alpha) \leq k$.*

Proof. Suppose firstly that α is k-local, and let $\sigma = (x_1, x_2, \ldots, x_n)$ be an optimal marking sequence of α. Consider the linear arrangement $L = (x_1, x_2, \ldots, x_n, \$, \#)$. We observe that $|\,\mathcal{C}(\{x_1, x_2, \ldots, x_n, \$\}, \{\#\})| = 2k$ and $|\,\mathcal{C}(\{x_1, x_2, \ldots, x_n\}, \{\$, \#\})| = 2$. Now consider a cut (K_1, K_2) with $K_1 = \{x_1, x_2, \ldots, x_i\}$ and $K_2 = \{x_{i+1}, \ldots, x_n, \$, \#\})$ for $1 \leq i < n$. Every edge $e \in \mathcal{C}(K_1, K_2)$ is of the form $\{x_j, x_h\}$ with $j \leq i < h$, or of the form $\{\alpha[1], \$\}$ or $\{\$, \alpha[|\alpha|]\}$. Consequently, every edge $e \in \mathcal{C}(K_1, K_2)$ corresponds to a unique factor $x_j x_h$ or $x_h x_j$ of α with $j \leq i < h$ and, after exactly the symbols $x_1, x_2, \ldots, x_i$ are marked, x_j is marked and x_h is not, or to a unique factor $\alpha[1]$ or $\alpha[|\alpha|]$ and, after exactly the symbols $x_1, x_2, \ldots, x_i$ are marked, $\alpha[1]$ or $\alpha[|\alpha|]$ is marked. Since there can be at most k marked blocks in α after marking the symbols $x_1, \ldots, x_i$, there are at most $2k$ such factors, which means that $|\,\mathcal{C}(K_1, K_2)| \leq 2k$. Thus $\mathsf{cw}(H_{\alpha,k}) \leq 2k$. Note that any linear arrangement must at some point separate the nodes \$ and #, meaning $\mathsf{cw}(H_{\alpha,k}) \geq 2k$, so we get that $\mathsf{cw}(H_{\alpha,k}) = 2k$.

Now suppose that the cutwidth of $H_{\alpha,k}$ is $2k$ and let L be an optimal linear arrangement witnessing this fact. Firstly, we note that L must either start with # followed by \$ (i.e., have the form $(\#, \$, \ldots)$) or end with # preceded by \$ (i.e., have the form $(\ldots, \$, \#)$. Otherwise, since $H_{\alpha,k}$ is connected, every cut separating \$ and # would be of size strictly greater than $2k$. Since a linear ordering and its mirror image have the same cutwidth, we may assume that the optimal linear arrangement has the form $L = (x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(n)}, \$, \#)$ for some permutation τ of $\{1, \ldots, n\}$. Let σ be the marking sequence $(x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(n)})$ of α induced by τ. Suppose, for contradiction, that for some i, with $1 \leq i < n$, after marking $x_{\tau(1)}, \ldots, x_{\tau(i)}$, we have $k' > k$ marked blocks. Furthermore, let $K_1 = \{x_{\tau(1)}, \ldots, x_{\tau(i)}\}$ and $K_2 = \{x_{\tau(i+1)}, \ldots, x_{\tau(n)}, \$, \#\}$. For every marked block $\alpha[s..t]$ that is not a prefix or a suffix of α, we have $\alpha[s], \alpha[t] \in K_1$ and $\alpha[s-1], \alpha[t+1] \in K_2$ and therefore $\{\alpha[s-1], \alpha[s]\}, \{\alpha[t], \alpha[t+1]\} \in \mathcal{C}(K_1, K_2)$. Moreover, for a marked prefix $\alpha[1..s]$, we have $\alpha[1], \alpha[s] \in K_1$ and $\$, \alpha[s+1] \in K_2$ and therefore $\{\alpha[1], \$\}, \{\alpha[s], \alpha[s+1]\} \in \mathcal{C}(K_1, K_2)$. Analogously, the existence of a marked suffix $\alpha[t..|\alpha|]$ leads to $\{\alpha[|\alpha|], \$\}, \{\alpha[t-1], \alpha[t]\} \in \mathcal{C}(K_1, K_2)$. Consequently, for each marked block, we have two unique edges in $\mathcal{C}(K_1, K_2)$, which implies $|\,\mathcal{C}(K_1, K_2)| \geq 2k' > 2k$. This contradicts the assumption that L is a witness that $H_{\alpha,k}$ has cutwidth $2k$. Thus, α must be k-local. □

Next we formally state how upper complexity bounds for Cutwidth carry over to Loc via the reduction above. In particular, we formulate this result to also cover fpt-algorithms with respect to the standard parameters $\mathsf{cw}(G)$ and $\mathsf{loc}(\alpha)$. The maximum degree in a multigraph G is bounded from above by $2\,\mathsf{cw}(G)$, so the number of nodes n and the number of edges h satisfy $h \leq n \cdot \mathsf{cw}(G)$.

Hence, we state the complexity in terms of n and $\mathsf{cw}(G)$ rather than with respect to h, which is the actual input size (assuming connected graphs).

Lemma 4.2. *If* MinCutwidth *(resp.* Cutwidth*) can be solved in* $\mathrm{O}(f(\mathsf{cw}(G), n))$ *time for a multigraph* G *with* n *vertices, then* MinLoc *(resp.,* Loc*) can be solved in* $\mathrm{O}(f(2\,\mathsf{loc}(\alpha), |\Sigma| + 2) + |\alpha|)$ *time for a word* α *over an alphabet* Σ*.*

Proof. We only show the claim for MinCutwidth; the case of Cutwidth follows immediately from Lemma 4.1. Our goal is to compute $\mathsf{loc}(\alpha)$ for the word α, i.e., the minimum k such that α is k-local. By Lemma 4.1, we get $\mathsf{cw}(H_{\alpha,k}) = 2k$ for $k \geq \mathsf{loc}(\alpha)$ and $\mathsf{cw}(H_{\alpha,k}) > 2k$ for $k < \mathsf{loc}(\alpha)$. Consider the multigraph H_α obtained by removing the vertices # and \$ from $H_{\alpha,i}$ (the result does not depend on $i \in \mathbb{N}$), and observe that $2\,\mathsf{loc}(\alpha) - 4 \leq \mathsf{cw}(H_\alpha) \leq 2\,\mathsf{loc}(\alpha)$. Indeed, if $\mathsf{cw}(H_\alpha) < 2\,\mathsf{loc}(\alpha) - 4$, we add the two missing nodes # and \$ (in this order) as a prefix to an optimal linear arrangement for H_α and get a linear arrangement of $H_{\alpha,\mathsf{loc}(\alpha)-1}$ of width $2\,\mathsf{loc}(\alpha)-2$, a contradiction.

Hence, in order to determine $\mathsf{loc}(\alpha)$, we proceed as follows: Compute $\ell = \mathsf{cw}(H_\alpha)$ and iterate over integers k, $\frac{\ell}{2} \leq k \leq \frac{\ell+4}{2}$, in increasing order, checking if $\mathsf{cw}(H_{\alpha,k}) = 2k$. The first value for which this equality holds equals $\mathsf{loc}(\alpha)$, and the marking sequence induced by the respective linear arrangement of $H_{\alpha,k}$ is an optimal one for α (as proved in Lemma 4.1). □

Next, we formally state and prove the approximation preserving properties of this reduction.

Lemma 4.3. *If there is an* $r(\mathsf{opt}, h)$*-approximation algorithm for* MinCutwidth *running in* $\mathrm{O}(f(h))$ *time for an input multigraph with* h *edges, then there is an* $(r(2\,\mathsf{opt}, |\alpha|)+\frac{1}{\mathsf{opt}})$*-approximation algorithm for* MinLoc *running in* $\mathrm{O}(f(|\alpha|) + |\alpha|)$ *time on an input word* α*.*

Proof. As already indicated in the proof of Lemma 4.1, for $k = \mathsf{loc}(\alpha)$, every linear arrangement for $H_{\alpha,k}$ naturally translates to a marking sequence for α. However, in an approximate linear arrangement, the vertices # and \$ do not have to be at the first (or last) positions. Still, the marking sequence corresponding to the linear arrangement L can have not more than $\frac{\mathsf{cw}(L)}{2} + 1$ marked blocks, since only suffix and prefix can be marked blocks which correspond to only one instead of two edges in a cut in $H_{\alpha,k}$. This observation remains valid if we do not include the extra vertices # and \$ in $H_{\alpha,k}$ in the reduction. Let H_α be the graph obtained from $H_{\alpha,k}$ (for some k) by removing the extra vertices # and \$ (observe that this also removes the dependence on k). Removing vertices only decreases the cutwidth, so Lemma 4.1 implies that $\mathsf{cw}(H_\alpha) \leq 2m^*(\alpha)$. Let α be an instance of MinLoc and $\mathcal{A}$ an $r(\mathsf{opt}, h)$-approximation for MinCutwidth on multigraphs. The approximation algorithm $\mathcal{A}$ run on H_α returns a linear arrangement $L = \mathcal{A}(H_\alpha)$ with $\mathsf{cw}(L) \leq r(\mathsf{opt}, h)\,\mathsf{cw}(H_\alpha)$. Let σ be the marking sequence corresponding to L, then $R(\alpha, \sigma) = \frac{\pi_\sigma(\alpha)}{m^*(\alpha)} \leq \frac{2}{\mathsf{cw}(H_\alpha)}(\frac{\mathsf{cw}(L)}{2} + 1) = \frac{\mathsf{cw}(L)}{\mathsf{cw}(H_\alpha)} + \frac{1}{m^*(\alpha)} = R(H_\alpha, L) + \frac{1}{m^*(\alpha)}$. The performance ratio $R(H_\alpha, L)$ is at most $r(\mathsf{opt}, h)$, where $h = |\alpha|$ is the number of edges in H_α. For the optimum value $k = m^*(\alpha)$, the cutwidth of $H_{\alpha,k}$ is at least $2k - 2$ and σ has performance ratio at most $r(2\,\mathsf{opt}, |\alpha|)$ (with respect to the optimum value k for MinLoc). The approximation procedure builds the graph H_α in $\mathrm{O}(|\Sigma|)$, runs $\mathcal{A}$ on H_α in $\mathrm{O}(f(|\alpha|))$ and translates the linear arrangement into a marking sequence σ in $\mathrm{O}(|\Sigma|)$. This gives an $(r(2\,\mathsf{opt}, |\alpha|) + \frac{1}{\mathsf{opt}})$-approximation for MinLoc running time in $\mathrm{O}(f(|\alpha|) + |\alpha|)$. □

4.2 Reducing Cutwidth to Locality Number

We next describe a reduction from Cutwidth to Loc. To this end, let $H = (V, E)$ be a connected multigraph, where V is the set of nodes and E the multiset of edges (for technical reasons, we assume $|V| \geq 2$). First, we construct the multigraph $H' = (V, E')$ obtained by duplicating every edge in H. As such, each node in H' has even degree, so we can fix some Eulerian cycle C (i.e., a cycle visiting each edge exactly once) in H', and, moreover, $\mathsf{cw}(H') = 2\,\mathsf{cw}(H)$. For each edge $e \in E'$, let α_e be the word over V that represents a traversal of the Eulerian path P obtained from C by deleting e. It is not important on which endpoint of the deleted edge e we start the traversal of the Eulerian path P (note that by introducing some order on V, we could easily make this choice unique).

Let us again clarify this reduction with an example. Let H be the graph shown on the left of Figure 5 (note that this is the same graph from Figure 2). By duplicating every edge in H,

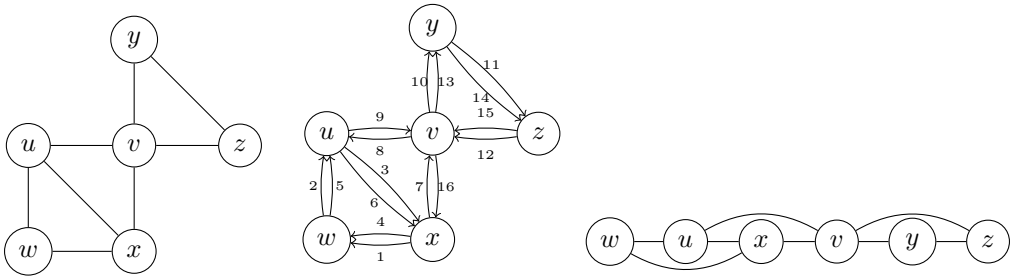

Figure 5: A graph H and its multigraph H' obtained by doubling the edges; the edge labels describe a Eulerian cycle that starts and ends in x. Deleting the edge (v, x) in this cycle yields the word $\alpha_{(v,x)} = xwuxwuxvuvyzvyzv$, which has an optimal marking sequence (w, u, x, v, y, z) with marking number 3, and, thus, induces an optimal linear arrangement of H with cutwidth 3.

we obtain the graph H' shown in the middle of Figure 5 (the edge directions and labels are not an explicit part of the reduction and shall serve illustrative purposes). A possible Eulerian cylce of H' that starts in vertex x is illustrated by the edge labels and the edge directions, i. e., the Eulerian cycle is $(x, w, u, x, w, u, x, v, u, v, y, z, v, y, z, v, x)$. Now splitting this cycle into a path by deleting its last edge $e = (v, x)$ results in a Eulerian path that corresponds to the word $\alpha_{(v,x)} = xwuxwuxvuvyzvyzv$.

The important property of the word α_e is that for every edge $\{x, y\}$ of H (except e), it contains two distinct size-2 factors that are xy- or yx-factors (for example, the original edge $\{x, w\}$ translates into two xw-factors, while the original edge $\{u, v\}$ translates into a vu-factor and a uv-factor). Consider the cuts of a fixed linear arrangement of H' from left to right and the marked versions of α_e with respect to the corresponding marking sequence. By construction, every boundary between a currently marked block and an adjacent unmarked block corresponds to a crossing edge of the current cut. This means that if there are ℓ marked blocks, then, depending on whether there is a marked prefix or suffix, the current cut must have size at least $2(\ell - 1)$ and at most 2ℓ. On the other hand, every crossing edge of the current cut (except e, if contained in the cut) is responsible for a marked symbol next to an unmarked one. This means that if the size of the current cut is 2ℓ (note that it must be even due to the duplication of edges), then there are ℓ marked blocks if no prefix or suffix is marked, there are $\ell + 1$ marked blocks if both a prefix and a suffix is marked, and if a prefix is marked but no suffix is marked (or the other way around), then in the current marked version there are $2\ell - 1$ boundaries between marked and unmarked blocks, and therefore the current cut contains $2\ell - 1$ edges different from e (the ones responsible for the $2\ell - 1$ boundaries between marked and unmarked blocks), and the additional edge e, which is not represented by any size-2 factor in α_e. Consequently, if H' has a cutwidth of $2k$ (which means that H has a cutwidth of k), then the locality number of α_e is either k or $k + 1$.

If $(v_1, v_2, \ldots, v_n)$ is a linear ordering for H with optimal cutwidth, then choosing any e that is adjacent to v_n will result in a word α_e for which, if marked according to H's optimal linear arrangement, the situation that both a prefix and suffix is marked can only happen at the end when all symbols are marked. This means that for such a choice of e, we have $\mathsf{loc}(\alpha_e) = \mathsf{cw}(H)$.

With respect to our example, we note that in fact the optimal linear arrangement shown on the right of Figure 5 has a cutwidth of 3, while the corresponding optimal marking sequence has a marking number of 3 for the word $\alpha_{\{v,x\}}$ (note that here the edge $\{v, x\}$ is not adjacent to the first or last vertex of the linear arrangement). We shall next provide a formal proof for these intuitive observations.

Lemma 4.4. *For any edge e in E', the word α_e satisfies $\mathsf{cw}(H) \leq \mathsf{loc}(\alpha_e) \leq \mathsf{cw}(H)+1$. Moreover, there is a vertex $v \in V$ such that $\mathsf{loc}(\alpha_e) = \mathsf{cw}(H)$ for every edge e incident to v.*

Proof. Let $k = \mathsf{cw}(H)$. Note that there is a natural bijection between the linear arrangements of H' and the marking sequences of the word α_e, since they both are essentially permutations of $\{1, 2, \ldots, n\}$, i. e., for a permutation τ of $\{1, 2, \ldots, n\}$, we can interpret $(x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(n)})$ both as the linear arrangement for H' and the a marking sequence of α_e *induced by* τ. In the following,

let τ be a permutation of $\{1, 2, \ldots, n\}$, let $i \in \{1, 2, \ldots, n-1\}$, $K_1 = \{x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(i)}\}$ and $K_2 = \{x_{\tau(i+1)}, \ldots, x_{\tau(n)}\}$, and let $\mathcal{C}(K_1, K_2) = 2\ell$ (note that since every edge has been duplicated, the size of every cut of H' is even).

Consider α_e after marking the letters $x_1, \ldots, x_{\tau(i)}$. For every marked block $\alpha[s..t]$ that is not a prefix or a suffix of α, we have $\alpha[s], \alpha[t] \in K_1$ and $\alpha[s-1], \alpha[t+1] \in K_2$ and therefore $\{\alpha[s-1], \alpha[s]\}, \{\alpha[t], \alpha[t+1]\} \in \mathcal{C}(K_1, K_2)$. Moreover, for a marked prefix $\alpha[1..s]$, we have $\alpha[s] \in K_1$ and $\alpha[s+1] \in K_2$ and therefore $\{\alpha[s], \alpha[s+1]\} \in \mathcal{C}(K_1, K_2)$. Analogously, the existence of a marked suffix $\alpha[t..|\alpha|]$ leads to $\{\alpha[t-1], \alpha[t]\} \in \mathcal{C}(K_1, K_2)$.

Conversely, for every edge in $\mathcal{C}(K_1, K_2)$, with the exception of e (if e is in $\mathcal{C}(K_1, K_2)$ at all), there is a unique length-2 factor $\alpha_e[p..p+1]$ of α_e such that either $\alpha_e[p]$ is marked and $\alpha_e[p+1]$ is unmarked, or vice-versa. Thus, if all marked blocks are internal, i.e., no marked block is a prefix or a suffix, then there are exactly ℓ marked blocks. Also, if both a prefix and a suffix occurs as a marked block, then we have $\ell + 1$ marked blocks. Finally, if a prefix occurs as a marked block, but no suffix, or vice-versa, then there are only ℓ marked blocks; note that in this case we must have $e \in \mathcal{C}(K_1, K_2)$. Since we consider all permutations, the arguments above are sufficient to conclude that, in our setting, each α_e has locality number either k or $k+1$.

Furthermore, consider a linear ordering $L = (x_{j_1}, \ldots, x_{j_n})$ of H' which is optimal, i.e., $|\mathcal{C}_L(i)| \leq 2k$. Note that if either the first or last letter of α_e is the last letter x_{j_n} to be marked according to the marking sequence induced by the linear ordering $(x_{j_1}, \ldots, x_{j_n})$, the case that both a suffix and prefix of α_e are marked cannot be reached until $i = n$ and the entire word is marked. Consequently, this would imply that α_e has locality number k. For any permutation of the linear ordering $(x_{j_1}, \ldots, x_{j_n})$, this holds for α_e where e is an edge adjacent to the node x_{j_n}, since the path P obtained by removing such an edge e from C must start or end with x_{j_n}. □

Again, we formally state and prove the approximation preserving properties of this reduction.

Lemma 4.5. *If there is an $r(\mathsf{opt}, |\alpha|)$-approximation algorithm for* MinLoc *running in $\mathrm{O}(f(|\alpha|))$ time on a word α, then there is an $r(\mathsf{opt}, h)$-approximation algorithm for* MinCutwidth *running in $\mathrm{O}(n(f(h) + h))$ time on a multigraph with n vertices and h edges.*

Proof. Let $G = (V, E)$ be an instance of MinCutwidth and $\mathcal{A}$ an $r(\mathsf{opt}, |\alpha|)$-approximation algorithm for MinLoc. By Lemma 4.4, there exists a vertex $v \in V$ such that $\mathsf{loc}(\alpha_e) = \mathsf{cw}(G)$ for any edge $e \in E$ adjacent to v. The approximation algorithm $\mathcal{A}$ hence returns on input α_e a marking sequence σ with $\pi_\sigma(\alpha_e) \leq r(\mathsf{opt}, |\alpha|)\, \mathsf{cw}(G)$.

In the proof of Lemma 4.4 it is further shown that any marking sequence σ for α_e translates to a linear arrangement L for G with $\mathsf{cw}(L) \leq \pi_\sigma(\alpha_e)$. The performance ratio of this linear arrangement is $R(G, L) = \frac{\mathsf{cw}(L)}{\mathsf{cw}(G)} \leq \frac{\pi_\sigma(\alpha_e)}{\mathsf{loc}(\alpha_e)} \leq R(\alpha_e, \sigma)$.

The procedure which, for each vertex $v \in V$, constructs α_e for some $e \in E$ adjacent to v in $\mathrm{O}(h)$, runs $\mathcal{A}$ in $\mathrm{O}(f(|\alpha_e|)) = \mathrm{O}(f(h))$ and checks the resulting linear arrangement in $\mathrm{O}(h)$ and returns the best linear arrangement among all $v \in V$, yields an $r(\mathsf{opt}, h)$-approximation for MinCutwidth on multigraphs in $\mathrm{O}(n(f(h) + h))$. □

4.3 Consequences of the Reductions

In the following, we discuss the lower and upper complexity bounds that we obtain from the reductions provided above. We first note that since Cutwidth is NP-complete, so is Loc. In particular, note that this answers one of the main questions left open in [15].

Theorem 4.6. Loc *is NP-complete (under Turing reductions), even if every symbol has at most* 3 *occurrences.*

Proof. Lemma 4.4 shows a polynomial time Turing reduction from Cutwidth to Loc. Indeed, given a (multi)graph H we construct in linear time the multigraph H' by duplicating its edges. H' has an Eulerian cycle, so, using Hierholzer's algorithm, we can compute such a cycle in linear time [32]. Let C be the computed Eulerian cycle. For each edge e of C construct, in linear time, the word α_e as described before Lemma 4.4. By Lemma 4.4 we get that $\mathsf{cw}(H) = \frac{\mathsf{cw}(H')}{2} = \min\{\mathsf{loc}(\alpha_e) \mid e \text{ edge of } C\}$. This completes the reduction, and, thus, as Cutwidth is NP-hard (see, e. g., [18]), we get that Loc is also NP-hard.

In order to show that the hardness holds even if every symbol has at most 3 occurrences, we first observe that in the reduction from Section 4.2, the number of occurrences of any symbol x in the constructed word α_e corresponds to the degree of the vertex x in the graph H. Hence, since Cutwidth is NP-complete already for graphs with maximum degree 3 (see [40]), it follows that Loc is NP-complete even if every symbol has at most 3 occurrences.

In order to show that Loc is in NP, let $\alpha \in \Sigma^*$ be an arbitrary word and $k \in \mathbb{N}$. We can guess a marking sequence σ for α in polynomial time, and then check in polynomial time whether its marking number $\pi_\sigma(\alpha)$ is less than or equal to k. □

Next, we formally state the positive fixed-parameter tractability results that Loc inherits from Cutwidth via the reduction from Section 4.1 (note that the fixed-parameter tractability of Loc was left as open problem in [15]).

Theorem 4.7. Loc *is fixed-parameter tractable if parameterised by* $|\Sigma|$*. Moreover, it can be solved in time and space* $O^*(2^{|\Sigma|})$*, or in* $O^*(4^{|\Sigma|})$ *time and polynomial space.*

Proof. In [12], the authors present algorithms for Cutwidth that run in $O^*(2^n)$ time and space, or in $O^*(4^n)$ time and polynomial space (where n is the number of vertices), and they also work for multigraphs.[2] Hence, Lemma 4.2 implies that Loc can be solved in $O^*(2^{|\Sigma|})$ time and space, or in $O^*(4^{|\Sigma|})$ time and polynomial space. □

Theorem 4.8. Loc *is fixed-parameter tractable if parameterised by* $\mathsf{loc}(\alpha)$*. Moreover, it can be solved with linear fpt-running-time* $g(\mathsf{loc}(\alpha))\,O(|\Sigma|)$*.*

Proof. The algorithm from [51] solves Cutwidth with linear fpt-running-time $g(\mathsf{cw}(G))\,O(n)$ (where n is the number of vertices). Hence, Lemma 4.2 implies that Loc can be solved with linear fpt-running-time $g(\mathsf{loc}(\alpha))\,O(|\Sigma|)$. □

The most natural parameters for Loc are the alphabet size $|\Sigma|$ and the standard parameter $\mathsf{loc}(\alpha)$ (recall that we have just seen that Loc is fixed-parameter tractable with respect to these two parameters). However, for string problems it is also common to investigate the parameterised complexity with respect to the maximum number of occurrences per symbols in the word α (we denote this parameter by $|\alpha|_{\mathsf{maxocc}}$). Theorem 4.6 already demonstrated that Loc is not fixed-parameter tractable with respect to $|\alpha|_{\mathsf{maxocc}}$ (unless P = NP). However, we only know that Loc stays NP-hard if $|\alpha|_{\mathsf{maxocc}}$ is bounded by a constant k with $k \geq 3$, and that the problem is trivial if $|\alpha|_{\mathsf{maxocc}}$ is bounded by 1 (in this case, the locality number is always 1). The complexity of Loc is open for the case where $|\alpha|_{\mathsf{maxocc}} \leq 2$.

Open Problem 4.9. Can Loc be solved in polynomial time if $|\alpha|_{\mathsf{maxocc}} \leq 2$?

If, on the other hand, the maximum number of occurrences per symbol in α is large in terms of α's length, i. e., we have that $|\alpha|_{\mathsf{maxocc}} = \Omega(\frac{|\alpha|}{\log(|\alpha|)})$, then Loc can be solved in polynomial time. Indeed, since $|\alpha|_{\mathsf{maxocc}} \geq \frac{|\alpha|}{|\Sigma|}$, we can conclude that $\frac{|\alpha|}{|\Sigma|} = \Omega(\frac{|\alpha|}{\log(|\alpha|)})$, which also means that $\log(|\alpha|) = \Omega(|\Sigma|)$. Consequently, $|\Sigma| = O(\log(|\alpha|))$, which means that Loc can be solved in polynomial time by using the $O^*(2^{|\Sigma|})$-time algorithm mentioned in Theorem 4.7.

We conclude this section by pointing out that Lemmas 4.3 and 4.5 imply some approximation results for MinLoc. However, we shall discuss approximation issues in greater detail in Section 5.

5 Locality and Pathwidth

One of the main results of this section is a reduction from the problem of computing the locality number of a word α to the probem of computing the pathwidth of a graph. This reduction, however, does not technically provide a reduction from the decision problem Loc to Pathwidth, since the constructed graph's pathwidth ranges between $\mathsf{loc}(\alpha)$ and $2\,\mathsf{loc}(\alpha)$, and therefore the reduction cannot be used to solve MinLoc exactly. The main purpose of this reduction is to carry over

[2] These algorithms actually support weighted graphs without any major modification and in the same complexity. In this setting, parallel edges connecting two vertices are replaced by a single "super-edge" whose weight is the number of parallel edges.

approximation results from MinPathwidth to MinLoc (also recall that exact and fpt-algorithms for MinLoc are obtained in Section 4 via a reduction to MinCutwidth). Hence, in this section we are mainly concerned with approximation algorithms.

Our strongest positive result about the approximation of the locality number will be derived from the reduction mentioned above (see Section 5.2). However, we shall first investigate in Section 5.1 the approximation performance of several obvious greedy strategies to compute the locality number (with "greedy strategies", we mean simple algorithmic strategies that build up a marking sequence from left to right by choosing the next symbol to be marked by some simple greedy rule). This is mainly motivated by two aspects. Firstly, ruling out simple strategies is a natural initial step in the search for approximation algorithms for a new problem. Secondly, due to the results of Section 4, the investigated greedy strategies for computing the locality number can also be interpreted as greedy strategies for computing the cutwidth of a graph. This may provide a new angle to approximating the cutwidth of a graph, i.e., some greedy strategies may only become apparent in the locality number point of view and are hard to see in the graph formulation of the problem. It may seem naive to expect new approximation results for cutwidth in this way, but, as mentioned in the introduction and as shall be discussed in detail in Section 6, approximating the cutwidth via approximation of the locality number may be beneficial for cutwidth approximation (although not by using simple greedy strategies, but the algorithm that follows from the reduction to computing the pathwidth).

Before presenting the main results of this section, let us briefly discuss some inapproximability results for MinLoc that directly follow from the reductions of Section 4 and known results about cutwidth approximation. Firstly, it is known that, assuming the Small Set Expansion Conjecture (denoted SSE; see [44]), there exists no constant-ratio approximation for MinCutwidth (see [52]). Consequently, approximating MinLoc within any constant factor is also SSE-hard. In particular, we point out that stronger inapproximability results for MinCutwidth are not known.

On certain graph classes, the SSE conjecture is equivalent to the Unique Games Conjecture [35] (see [44, 45]), which, at its turn, was used to show that many approximation algorithms are tight (see [36]) and is considered a major conjecture in inapproximability. However, some works seem to provide evidence that could lead to a refutation of SSE; see [3, 6, 31]. In this context, our negative result of Section 5.1 can also be interpreted as a series of unconditional results which state that multiple natural greedy strategies for computing the locality number (and their equivalents for computing the cutwidth) do not provide low-ratio approximations of MinLoc (or MinCutwidth, respectively).

5.1 Greedy Strategies

Since a marking sequence is just a linear arrangement of the symbols of the input word, computing marking sequences seems to be well tailored to greedy algorithms: until all symbols are marked, we choose an unmarked symbol according to some greedy strategy and mark it. Unfortunately, we can formally show that many natural candidates for greedy strategies fail to yield promising approximation algorithms (and are therefore also not helpful for cutwidth approximation).

For a systematic investigation, we shall now define our *basic greedy strategies*:

FewOcc	Among all unmarked symbols, choose one with a smallest number of occurrences.
ManyOcc	Among all unmarked symbols, choose one with a largest number of occurrences.
FewBlocks	Among all unmarked symbols, choose one that, after marking it, results in the smallest total number of marked blocks.
LeftRight	Among all unmarked symbols, choose the one with the leftmost occurrence.

These strategies are – except for LeftRight – nondeterministic, since there are in general several valid choices of the next symbol to mark. However, we will show poor performances for these strategies independent of the nondeterministic choices (i. e., the approximation ratio is bad for every possible nondeterministic choices), which are stronger negative results. We make the convention that all strategies – except, of course, LeftRight – can choose any symbol as the initially marked symbol, which is justified by the fact that, in terms of running-time, we could afford to try out every possible choice of the first symbol.

In the following, for every greedy strategy S and for every word α, let $\texttt{GREEDY}_S(\alpha)$ be the

optimal marking number over all marking sequences that can be obtained by strategy S. For every word α let $\psi_S(\alpha) = \frac{\texttt{GREEDY}_S(\alpha)}{\mathsf{loc}(\alpha)}$, and for every $\ell \in \mathbb{N}$, let $\psi_S(\ell) = \max\{\psi_S(\alpha) \mid |\alpha| = \ell\}$; the function $\psi_S : \mathbb{N} \to \mathbb{N}$ is called the *approximation performance of strategy* S. Our negative results will be as follows. For every strategy S and for every $\ell \in \mathbb{N}$, we show that there is a constant c and a length-ℓ word α, such that $\mathsf{loc}(\alpha) = c$, while $\texttt{GREEDY}_S(\alpha) = \Omega(\ell)$. Note that this means that the approximation performance $\psi_S(\ell)$ of strategy S is linear.

We first investigate strategies FewOcc and FewBlocks. For every $\ell \geq 2$, let

$$\alpha = (x_1 x_2 \dots x_\ell)^2 x_1 \beta_1 x_2 \beta_2 x_3 \beta_3 \dots \beta_{\ell-1} x_\ell \,,$$

where, for every i, $1 \leq i \leq \ell - 1$, $\beta_i = (y_{2i-1} y_{2i})^4$.

For example, if $\ell = 4$, then we obtain $\alpha = (x_1x_2x_3x_4)^2 x_1 (y_1y_2)^4 x_2 (y_3y_4)^4 x_3 (y_5y_6)^4 x_4$. It can be easily seen that $|\alpha| = 11\ell - 8 = \mathrm{O}(\ell)$.

Lemma 5.1. *$\psi_{\mathsf{FewOcc}}(\alpha) \geq \frac{\ell-1}{6}$ and $\psi_{\mathsf{FewBlocks}}(\alpha) \geq \frac{\ell-1}{6}$.*

Proof. We first observe that $(x_1, y_1, y_2, x_2, y_3, y_4, x_3, y_5, y_6, \dots)$ is an optimal marking sequence which shows that $\mathsf{loc}(\alpha) = 6$. Next, we consider how the strategies FewOcc and FewBlocks can mark α. If the first marked symbol is some x_i, then FewOcc would next mark all remaining x_j, $j \neq i$, in some order, since each symbols x_i has fewer occurrences than any of the symbols y_j, and FewBlocks would next mark all remaining x_j, $j \neq i$, since these can be marked in such an order that per new marking we increase the number of new blocks only by one, while marking some y_j would increase the number of marked blocks by three. This leads to at least ℓ marked blocks. If, on the other hand, some y_{2j-1} or y_{2j} is marked first, then FewOcc marks some x_i next and then all remaining $x_{i'}$ as before, while FewBlocks would mark the remaining symbol of y_{2i-1} or y_{2i} (because this is the only choice that does not increase the number of marked blocks) and then all x_i in some order that produces a minimal number of new marked blocks. This results in at least $\ell - 1$ marked blocks. Thus, $\psi_{\mathsf{FewOcc}}(\alpha) \geq \frac{\ell-1}{6}$ and $\psi_{\mathsf{FewBlocks}}(\alpha) \geq \frac{\ell-1}{6}$. □

Next, we consider the strategy ManyOcc. For every $\ell \geq 2$, let

$$\gamma = x_1 x_2 \dots x_\ell x_1 y_1 x_2 y_2 x_3 y_3 \dots y_{\ell-1} x_\ell \,.$$

For example, if $\ell = 5$, then we obtain $\gamma = x_1x_2x_3x_4x_5x_1y_1x_2y_2x_3y_3x_4y_4x_5$. It can be easily seen that $|\gamma| = 3\ell - 1 = \mathrm{O}(\ell)$.

Lemma 5.2. *$\psi_{\mathsf{ManyOcc}}(\gamma) \geq \frac{\ell-1}{2}$.*

Proof. We first observe that $(x_1, y_1, x_2, y_2, x_3, y_3, \dots)$ is an optimal marking sequence which shows that $\mathsf{loc}(\gamma) = 2$. If ManyOcc marks some x_i first, then it will mark all remaining x_j next, since each of these symbols have 2 occurrences. This results in ℓ marked blocks. If, on the other hand, the first symbol is some y_j, then again the symbols x_i have the most occurrences and are therefore marked next in some order. This leads to $\ell - 1$ marked blocks. Thus, $\psi_{\mathsf{ManyOcc}}(\gamma) \geq \frac{\ell-1}{2}$. □

Finally, we consider the strategy LeftRight. For every even number $\ell \geq 2$, let

$$\delta = x_1 x_2 \dots x_\ell x_1 x_\ell x_2 x_{\ell-1} x_3 x_{\ell-2} \dots x_{\frac{\ell}{2}} x_{\frac{\ell}{2}+1} \,.$$

For example, if $\ell = 6$, then we obtain $\delta = x_1x_2x_3x_4x_5x_6x_1x_6x_2x_5x_3x_4$. It can be easily seen that $|\delta| = 2\ell = \mathrm{O}(\ell)$.

Lemma 5.3. *$\psi_{\mathsf{LeftRight}}(\delta) \geq \frac{\ell}{4}$.*

Proof. We first observe that $\mathsf{loc}(\delta) = 2$, which is witnessed by the marking sequence

$$(x_1, x_\ell, x_2, x_{\ell-1}, x_3, x_{\ell-2}, \dots, x_{\frac{\ell}{2}}, x_{\frac{\ell}{2}+1})$$

(note that this marking sequence maintains a marked prefix and one additional marked internal factor starting with $x_\ell x_1 x_\ell$, which is alternately extended to both sides). Now assume that the strategy LeftRight marks some symbol x_i. If $i \leq \frac{\ell}{2}$, then it marks next all the symbol $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_{\frac{\ell}{2}}$, which results in $\frac{\ell}{2} + 1$ marked blocks. If, on the other hand, $i > \frac{\ell}{2}$, then symbols $x_1, \dots, x_{\frac{\ell}{2}}$ are marked next, which leads to at least $\frac{\ell}{2}$ marked blocks. Thus $\psi_{\mathsf{LeftRight}}(\delta) \geq \frac{\ell}{4}$. □

In the following, we investigate another aspect of greedy strategies. Any symbol that is marked next in a marking sequence can have *isolated* occurrences (i. e., occurrences that are not adjacent to any marked block) and *block-extending* occurrences (i. e., occurrences with at least one adjacent marked symbol). Each isolated occurrence results in a new marked block, while each block-extending occurrence just extends an already existing marked block, and potentially may even combine two marked blocks and therefore may decrease the overall number of marked blocks. Therefore, marking a symbol when it only has isolated occurrences causes the maximum number of marked blocks that can ever be contributed by this symbol, and therefore this seems to be the worst time to mark this symbol. Hence, in terms of a greedy strategy, it seems reasonable to only mark symbols if they also have block-extending occurrence (obviously, this is not possible for the initially marked symbol).

We call a marking sequence σ for a word α *block-extending*, if every symbol that is marked except the first one has at least one block-extending occurrence. This definition leads to the general combinatorial question of whether every word has an optimal marking sequence that is block-extending, or whether the seemingly bad choices of marking a symbol that has only isolated occurrences (and that is not the first symbol) is necessary for optimal marking sequences. We answer this question in the negative.

For every even number $\ell \geq 2$, let $\beta = x_1 y x_2 y x_3 y \dots x_\ell y$.

Proposition 5.4. *$\mathsf{loc}(\beta) = \frac{2}{\ell}$ and, for every block-extending marking sequence σ for β, we have $\pi_\sigma(\beta) \geq \ell - 1$.*

Proof. For the sake of convenience, let $\ell = 2k$ for some $k \geq 1$. Let σ be any block-extending marking sequence for α. If σ marks y first, then we have $2k$ marked blocks and if some x_i, $1 \leq i \leq 2k$, is marked first, then y is marked next, which leads to $2k-1$ marked blocks. Thus, $\pi_\sigma(\beta) \geq 2k - 1$. On the other hand, we can proceed as follows. We first mark the k symbols $x_2, x_3, \dots, x_{k+1}$, which leads to k marked blocks (and which is a marking sequence that is *not* block extending). Then we mark y, which joins all the previously marked blocks into one marked block and turns $k-1$ occurrences of y into new individual marked blocks (i. e., the $k-2$ occurrences of y between the symbols $x_{k+2}, x_{k+3}, \dots, x_{2k}$ and the single occurrence of y after x_{2k}). Thus, there are still k marked blocks, and from now on marking the rest of the symbols only decreases the number of marked blocks. Consequently, $\mathsf{loc}(\beta) \leq k$. Moreover, after any marking sequence has marked symbol y, there are $2k$ marked occurrences of symbol y. If these marked occurrences form at least k marked blocks, the overall marking number of the marking sequence is at least k. If they form at most $k-1$ marked blocks, then at least $k+1$ of the symbols x_i must be marked as well, and since these symbols were marked before marking y, they have formed at least $k+1$ marked blocks before marking y. This means that the overall marking number is at least $k+1$. This shows that $\mathsf{loc}(\beta) \geq k$, and therefore $\mathsf{loc}(\beta) = k$. □

This proposition points out that even simple words can have only optimal marking sequences that are not block-extending. In terms of greedy strategies however, Proposition 5.4 only shows a lower bound of roughly 2 for the approximation ratio of any greedy algorithm that employs some block-extending greedy strategy (since the lower bound applies to *every* marking sequence that is block-extending). We note that the requirement of marking in a block-extending way does not specify which one of the possible block-extending symbols should be marked; trying out all of them is obviously too costly. In order to further investigate block-extending greedy strategies, we therefore couple the block-extending requirement with other greedy strategies, e. g., for the strategies $S \in \{\mathsf{FewOcc}, \mathsf{ManyOcc}, \mathsf{FewBlocks}, \mathsf{LeftRight}\}$ from above, we denote by $\mathsf{BE}-S$ the greedy strategy which only marks block-extending symbols (except the first one) and chooses among possible block-extending symbols according to strategy S (note that these strategies are still non-deterministic in the sense of how S is a non-deterministic strategy). More precisely, the strategies $\mathsf{BE}-S$ are defined by replacing "unmarked symbols" by "unmarked symbols that have at least one block-extending occurrence" in the descriptions of the basic strategies from above.

We observe that for $S \in \{\mathsf{FewOcc}, \mathsf{ManyOcc}, \mathsf{FewBlocks}, \mathsf{LeftRight}\}$ the strategy $\mathsf{BE}-S$ is not covered by S, i. e., the set of marking sequences for a word that can be obtained by S is not necessarily a superset of the marking sequences that satisfy $\mathsf{BE}-S$. For example, an unmarked symbol that has a maximum (or minimum) number of occurrences among all block-extending

symbols, might not have a maximum (or minimum, respectively) number of occurrences among all unmarked symbols. Therefore, the lower bounds form Lemmas 5.1, 5.2 and 5.3 do not carry over automatically. Nevertheless, $\mathsf{BE}-S$ behaves more or less in the same way as S on the witness words α, γ and δ defined above, which yields the following.

Lemma 5.5. *Let the words α, γ and δ be defined as above. Then*

- $\psi_{\mathit{BE-FewOcc}}(\alpha) \geq \frac{\ell-1}{6}$,
- $\psi_{\mathit{BE-FewBlocks}}(\alpha) \geq \frac{\ell-1}{6}$,
- $\psi_{\mathit{BE-ManyOcc}}(\gamma) \geq \frac{\ell-1}{6}$,
- $\psi_{\mathit{BE-LeftRight}}(\delta) \geq \frac{\ell}{4}$.

Proof. Optimal marking sequences for the words and the lengths of these words have been discussed above, so it only remains to show that no $\mathsf{BE}-S$ with $S \in \{\mathsf{FewOcc}, \mathsf{ManyOcc}, \mathsf{FewBlocks}, \mathsf{LeftRight}\}$ can produce better marking sequences.

We first discuss $\mathsf{BE}-\mathsf{FewOcc}$ and $\mathsf{BE}-\mathsf{FewBlocks}$ together. If the initially marked symbol is some x_i, then both $\mathsf{BE}-\mathsf{FewOcc}$ and $\mathsf{BE}-\mathsf{FewBlocks}$ would next mark all remaining x_j, $j \neq i$. The only difference to strategies FewOcc and $\mathsf{FewBlocks}$ is that $\mathsf{BE}-\mathsf{FewOcc}$ and $\mathsf{BE}-\mathsf{FewBlocks}$ must mark these symbols such that always a block-extending symbol is marked next, i. e., we must mark in such a way there are always at most 2 marked blocks in the prefix $(x_1x_2\dots x_\ell)^2$. This leads to at least ℓ marked blocks. If some y_{2i-1} or y_{2i} is marked first, then $\mathsf{BE}-\mathsf{FewOcc}$ marks x_i or x_{i+1} next (depending on whether y_{2i-1} or y_{2i} is marked first) and then all remaining x_j as before, while $\mathsf{BE}-\mathsf{FewBlocks}$ would mark the remaining symbol of y_{2i-1} or y_{2i} and then all x_j. This results in at least $\ell-1$ marked blocks. Thus, $\psi_{\mathsf{BE}-\mathsf{FewOcc}}(\alpha) \geq \frac{\ell-1}{6}$ and $\psi_{\mathsf{BE}-\mathsf{FewBlocks}}(\alpha) \geq \frac{\ell-1}{6}$.

Next, we consider $\mathsf{BE}-\mathsf{ManyOcc}$. It can be easily seen that no matter whether we initially mark some x_i or some y_i, just like $\mathsf{ManyOcc}$ the strategy $\mathsf{BE}-\mathsf{ManyOcc}$ will mark all remaining x_i next (these symbols have a maximum number of occurrences and two of them must be block-extending). Thus, $\mathsf{BE}-\mathsf{ManyOcc}$ necessarily produces ℓ or $\ell-1$ marked blocks, and therefore $\psi_{\mathsf{BE}-\mathsf{ManyOcc}}(\gamma) \geq \frac{\ell-1}{6}$.

Finally, we consider $\mathsf{BE}-\mathsf{LeftRight}$. This strategy behaves similar to $\mathsf{LeftRight}$, but we have to argue a bit more carefully. Assume that x_i is the first symbol marked by $\mathsf{BE}-\mathsf{LeftRight}$. If $i \leq \frac{\ell}{2}$, then we mark next x_{i-1}, then x_{i-2} and so on, until all $x_1, x_2, \dots, x_i$ are marked. Then the symbols $x_{i+1}, x_{i+2}, \dots, x_{\frac{\ell}{2}}$ are marked, which leads to $\frac{\ell}{2}+1$ marked blocks. If, on the other hand, $i = \frac{\ell}{2}+j$ with $j \geq 1$, then the next marked symbol will be $x_{\frac{\ell}{2}-(j-1)}$. Then, as before, we will mark $x_{\frac{\ell}{2}-(j-1)-1}, x_{\frac{\ell}{2}-(j-1)-2}, \dots$ until all $x_1, x_2, \dots, x_{\frac{\ell}{2}-(j-1)}$ are marked, and then $x_{\frac{\ell}{2}-(j-1)+1}, \dots, x_{\frac{\ell}{2}}$ are marked. This results in $\frac{\ell}{2}$ marked blocks. Thus, $\psi_{\mathsf{BE}-\mathsf{LeftRight}}(\delta) \geq \frac{\ell}{4}$. □

If we are concerned with block-extending greedy strategies, then it is natural to choose among the block-extending symbols according to the number of their block-extending or isolated occurrences. This motivates the following two strategies:

BE-1	Among all block-extending symbols, choose one that has the most block-extending occurrences.
BE-2	Among all block-extending symbols, choose one for which $\frac{\#\text{block-extending occ.}}{\#\text{occ.}}$ is maximal.

Unfortunately, the witness word α also shows poor approximation ratios for these strategies.

Lemma 5.6. $\psi_{\mathit{BE}\text{-}1}(\alpha) \geq \frac{\ell-1}{6}$ *and* $\psi_{\mathit{BE}\text{-}2}(\alpha) \geq \frac{\ell-1}{6}$.

Proof. Again, we first note that the optimal marking sequence for α and the length $|\alpha|$ have already been discussed above. If we first mark a symbol x_i, then, among all symbols extending a marked block, i. e., symbols x_{i-1}, x_{i+1}, y_{2i-1} and y_{2i+1}, the symbols x_{i-1} and x_{i+1} each have 3 occurrences in total, two of which are block-extending, whereas the symbols y_{2i-1} and y_{2i+1} each have 4 occurrences, only one of which is block-extending. Consequently, both BE-1 and BE-2 chose

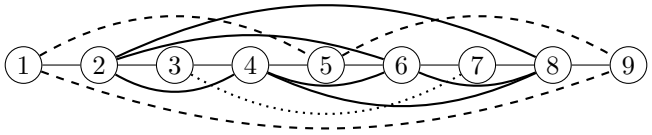

Figure 6: The graph G_α for $\alpha = \texttt{cabacabac}$; the three cliques are drawn with different edge-types.

either x_{i-1} or x_{i+1} next. This situation does not change until all x_i are marked, which leads to ℓ marked blocks. If, on the other hand, some y_{2i-1} or y_{2i} is marked first, then we mark next the remaining symbol y_{2i-1} or y_{2i} such that β_i is completely marked (this is due to the fact that this symbol is the only block-extending one). Next, x_i and x_{i+1} are marked, in some order (again, this is enforced by the fact that we can only mark block-extending symbols), which brings us back to the situation described above which leads to the marking of all remaining x_j, leading to $\ell - 1$ marked blocks. Consequently, $\psi_{\mathsf{BE\text{-}1}}(\alpha) \geq \frac{\ell-1}{6}$ and $\psi_{\mathsf{BE\text{-}2}}(\alpha) \geq \frac{\ell-1}{6}$. □

5.2 Reducing Locality Number to Pathwidth

In the following, we obtain an approximation algorithm for the locality number by reducing it to the problem of computing the pathwidth of a graph. To this end, we first describe another way of how a word can be represented by a graph. Recall that the reduction to cutwidth from Section 4 also transforms words into graphs. The main difference is that the reduction from Section 4 turns every symbol from the alphabet into an individual vertex of the graph (thus, producing a graph with $\mathrm{O}(|\Sigma|)$ vertices), while the reduction to pathwidth will use a vertex per position of the word α, i. e., $|\alpha|$ individual vertices. In the reduction from Section 4 the information of the actual occurrences of the symbols in the word is encoded by the edges (in particular, the length $|\alpha|$ is represented by the number of edges), while in the following reduction the alphabet is encoded by connecting the vertices that correspond to positions of the same symbol to cliques in the graph (in particular, the number of edges may range between $|\alpha|$ and $|\alpha|^2$). We proceed with a formal definition and an example.

For a word α, the graph $G_\alpha = (V_\alpha, E_\alpha)$ is defined by $V_\alpha = \{1, 2, \ldots, |\alpha|\}$ and $E_\alpha = \{\{i, i+1\} \mid 1 \leq i \leq |\alpha| - 1\} \cup \{\{i, j\} \mid \{i, j\} \subseteq \mathsf{ps}_x(\alpha) \text{ for some } x \in \mathsf{alph}(\alpha)\}$. Intuitively, G_α is a length-$|\alpha|$ path (due to the edges $\{\{i, i+1\} \mid 1 \leq i \leq |\alpha| - 1\}$), and, additionally, we add edges such that every set $\mathsf{ps}_x(\alpha)$, $x \in \mathsf{alph}(\alpha)$, is a clique.

Let us discuss a brief example. Let $\alpha = \texttt{cabacabac}$ be a word of length 9 over alphabet $\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$. The graph G_α has therefore vertices $\{1, 2, \ldots, 9\}$, which are connected into a path from vertex 1 to vertex 9. In addition, $\mathsf{ps}_\mathsf{a}(\alpha) = \{2, 4, 6, 8\}$ forms a clique, $\mathsf{ps}_\mathsf{b}(\alpha) = \{3, 7\}$ forms a clique, and $\mathsf{ps}_\mathsf{b}(\alpha) = \{1, 5, 9\}$ forms a clique. See Figure 6 for an illustration.

We use G_α as a unique graph representation for words and whenever we talk about a path decomposition for α, we actually refer to a path decomposition of G_α. Recall that we consider path-decompositions as certain marking schemes, which we called pd-marking schemes (see Section 2.3 and Figure 3). Since G_α has the positions of α as its vertices, the pd-marking scheme behind a path decomposition (and its respective terminology) directly translates to a marking scheme of the positions of α.

A very similar reduction has been used in [47] in order to prove that certain structural restrictions of patterns with variables lead to polynomial-time cases of the corresponding matching problem. The reduction from [47] is more general in the sense that it does not require the vertices of $\mathsf{ps}_x(\alpha)$ to be a clique, but only requires that these vertices form a connected component (if we do not count the "path-edges" $\{\{i, i+1\} \mid 1 \leq i \leq |\alpha| - 1\}$).

The main property of this reduction is that the pathwidth of G_α ranges between $\mathsf{loc}(\alpha)$ and $2\,\mathsf{loc}(\alpha)$.

Lemma 5.7. *Let α be a word with $|\alpha| \geq 2$. Then $\mathsf{loc}(\alpha) \leq \mathsf{pw}(G_\alpha) \leq 2\,\mathsf{loc}(\alpha)$. Further, given a path decomposition Q for G_α, a marking sequence σ for α with $\pi_\sigma(\alpha) \leq \mathsf{w}(Q)$ can be constructed in time $\mathrm{O}(|\alpha|)$.*

We defer the somewhat technical proof of Lemma 5.7 to the end of this section, and first discuss the consequences and some further properties of the reduction.

A rather simple observation is that the statement of Lemma 5.7 is in fact not true for words α of size 1, since then $\mathsf{loc}(\alpha) = 1$ and $\mathsf{pw}(G_\alpha) = 0$.

Intuitively speaking, every marked block in an optimal marking sequence for α accounts for one unit of the quantity $\mathsf{loc}(\alpha)$, while in an optimal path decomposition of G_α, any marked block is represented by two `active` vertices (i. e., vertices that are in the current bag, see the terminology introduced in Section 2.3). This explains why $\mathsf{pw}(G_\alpha)$ can be twice as large as $\mathsf{loc}(\alpha)$; on the other hand, that $\mathsf{pw}(G_\alpha)$ can be strictly smaller than $2\,\mathsf{loc}(\alpha)$ is due to the fact that every marked block of size 1 is actually represented by only 1 `active` vertex, instead of two. We can formally show that there are rather simple example words α and β that reach the extremes of $2\,\mathsf{loc}(\alpha) = \mathsf{pw}(G_\alpha)$ and $\mathsf{loc}(\beta) = \mathsf{pw}(G_\beta)$, i.e., the bounds of 5.7 are tight. The proof of the following proposition also serves as an introduction to path-decompositions for the graph representation G_α of words (and our use of the terminology explained in Section 2.3), and therefore as a preparation for the proof of Lemma 5.7.

Proposition 5.8. *Let $\alpha = (x_1x_2 \dots x_nx_{n-1} \dots x_2)^k x_1$ with $n \geq 3$, and let $\beta = (x_1x_2)^k$. Then we have $\mathsf{loc}(\alpha) = k$ and $\mathsf{pw}(G_\alpha) = 2k$, and $\mathsf{loc}(\beta) = \mathsf{pw}(G_\beta) = k$.*

Proof. We start with proving the first statement and first observe that $\mathsf{loc}(\alpha) \leq k$ due to the marking sequence $x_n, x_{n-1}, \dots, x_1$. In order to show $\mathsf{pw}(G_\alpha) \geq 2k$, we first observe that, for every $i \in \{2, \dots, n-1\}$, $\mathsf{ps}_{x_i}(\alpha)$ is a clique of size $2k$ in G_α, which implies that every path-decomposition Q (interpreted as a pd-marking scheme) for G_α reaches a step where all $2k$ vertices of $\mathsf{ps}_{x_i}(\alpha)$ are `active`. Now let Q be a path-decomposition for G_α (interpreted as a pd-marking scheme), let $i \in \{2, \dots, n-1\}$ be such that all $\mathsf{ps}_{x_i}(\alpha)$ are first set to `active`, i. e., when all vertices $\mathsf{ps}_{x_i}(\alpha)$ are `active` for the first time, then in every $\mathsf{ps}_{x_j}(\alpha)$, $j \in \{2, \dots, n-1\} \setminus \{i\}$, there is at least one `open` vertex (in particular, no vertex from any ps_{x_j}, $2 \leq j \leq n-1$, is `closed`). Moreover, in the following we consider the earliest point of Q, where all $\mathsf{ps}_{x_i}(\alpha)$ are `active`.

If, at this point, there is some additional `active` vertex, then there are $2k+1$ `active` vertices; thus, in the following we assume that there are no other `active` vertices. If there is also no `closed` vertex, then all other vertices are `open`, which means that every vertex from $\mathsf{ps}_{x_i}(\alpha)$ has at least one adjacent `open` vertex and therefore we have to set an `open` vertex to `active`, before we can set a vertex from $\mathsf{ps}_{x_i}(\alpha)$ to `closed`; this leads to at least $2k+1$ `active` vertices. It remains to consider the case where there is some `closed` vertex j. This means that all vertices of $\mathsf{ps}_{\alpha[j]}(\alpha)$ are `closed`, which implies that $j \in \mathsf{ps}_{x_1}(\alpha) \cup \mathsf{ps}_{x_n}(\alpha)$. We first consider the case $j \in \mathsf{ps}_{x_1}(\alpha)$. Since every vertex from $\mathsf{ps}_{x_2}(\alpha)$ is adjacent to some vertex from $\mathsf{ps}_{x_1}(\alpha)$, we can conclude that all vertices from $\mathsf{ps}_{x_2}(\alpha)$ are `active`, i. e., $i = 2$. The assumption $j \in \mathsf{ps}_{x_n}(\alpha)$ analogously leads to the situation that $i = n-1$. Consequently, all $2k$ vertices from $\mathsf{ps}_{x_i}(\alpha)$ are `active`, either $\mathsf{ps}_{x_1}(\alpha)$ are all `closed` or $\mathsf{ps}_{x_n}(\alpha)$ are all `closed`, and all other vertices are `open`. In both of these cases, every vertex from $\mathsf{ps}_{x_i}(\alpha)$ has at least one adjacent `open` vertex, which, as before, means that we have to set an `open` vertex to `active`, before we can set a vertex from $\mathsf{ps}_{x_i}(\alpha)$ to `closed`; this, as well, leads to at least $2k+1$ `active` vertices. Consequently, $\mathsf{w}(Q) \geq 2k$, and, with Lemma 5.7, we can conclude $\mathsf{pw}(G_\alpha) = 2k$.

With respect to the second statement, we first note that any marking sequence for β leads to k marked blocks, which implies $\mathsf{loc}(\beta) = k$. Moreover, a pd-marking scheme Q with $\mathsf{w}(Q) = k$ can be easily constructed as follows. First, we set all positions of $\mathsf{ps}_{x_1}(\beta)$ to `active`. Then we set position 2 to `active`, position 1 to `closed`, position 4 to `active`, position 3 to `closed` and so on, until all positions of $\mathsf{ps}_{x_2}(\beta)$ are `active` and all positions of $\mathsf{ps}_{x_1}(\beta)$ are `closed`. Finally, the positions of $\mathsf{ps}_{x_2}(\beta)$ are set to `closed`. There are at most $k+1$ positions `active` at the same time; thus, $\mathsf{w}(Q) = k$ and therefore $\mathsf{pw}(G_\beta) \leq k$. Together with Lemma 5.7, this implies $\mathsf{loc}(\beta) = \mathsf{pw}(G_\beta) = k$. □

As explained at the beginning of this section, the construction of a graph G_α from a word α does not reduce the decision problem Loc to Pathwidth (since $\mathsf{pw}(G_\alpha)$ lies between $\mathsf{loc}(\alpha)$ and $2\,\mathsf{loc}(\alpha)$); its main purpose is to obtain approximation results, which is formally stated by the next lemma.

Lemma 5.9. *If* MinPathwidth *admits an* $O(f(n))$*-time* $r(\mathsf{opt}, n)$*-approximation algorithm, then* MinLoc *admits an* $O(f(|\alpha|) + |\alpha|^2)$*-time* $2r(2\,\mathsf{opt}, |\alpha|)$*-approximation algorithm.*

Proof. Let α be an instance of MinLoc and $\mathcal{A}$ an $r(\mathsf{opt}, n)$-approximation for MinPathwidth. By Lemma 5.7, it follows that $\mathsf{pw}(G_\alpha) \leq 2m^*(\alpha)$.

Let Q be the path decomposition computed by $\mathcal{A}$ on G_α and σ be the corresponding marking sequence constructed with Lemma 5.7. With the inequality $m^*(\alpha) \geq \frac{1}{2}\,\mathsf{pw}(G_\alpha)$, the performance ratio of σ can be bounded by $R(\alpha, \sigma) = \frac{\pi_\sigma(\alpha)}{m^*(\alpha)} \leq \frac{2}{\mathsf{pw}(G_\alpha)}\,\mathsf{w}(Q) \leq 2R(G_\alpha, Q)$. With $R(G_\alpha, Q) \leq r(\mathsf{cw}(G_\alpha), n)$ from the approximation ratio of $\mathcal{A}$, $n = |\alpha|$ from the construction of G_α, and $\mathsf{pw}(G_\alpha) \leq 2m^*(\alpha)$ from Lemma 5.7, the claimed bound of $2r(2\,\mathsf{opt}, |\alpha|)$ on the approximation ratio follows. The approximation procedure to compute σ, creates G_α in $O(|\alpha|^2)$, runs $\mathcal{A}$ in $O(f(|\alpha|))$ and translates the path-decomposition Q into σ in $O(|\alpha|)$, which takes an overall running time in $O(f(|\alpha|) + |\alpha|^2)$. □

Consequently, approximation algorithms for MinPathwidth carry over to MinLoc. For example, the $O(\sqrt{\log(\mathsf{opt})}\log(n))$-approximation algorithm for MinPathwidth from [21] implies the following.

Theorem 5.10. *There is an* $O(\sqrt{\log(\mathsf{opt})}\log(n))$*-approximation algorithm for* MinLoc*.*

Another consequence that is worth mentioning is due to the fact that an optimal path decomposition can be computed faster than $O^*(2^n)$. More precisely, it is shown in [50] that for computing path decompositions, there is an exact algorithm with running time $O^*((1.9657)^n)$, and even an additive approximation algorithm with running time $O^*((1.89)^n)$. Consequently, there is a 2-approximation algorithm for MinLoc with running time $O^*((1.9657)^n)$ and an asymptotic 2-approximation algorithm with running time $O^*((1.89)^n)$ for MinLoc.

Many existing algorithms constructing path decompositions are of theoretical interest only, and this disadvantage carries over to the possible algorithms computing the locality number or cutwidth (see Section 6) based on them. However, the reduction of 5.7 is also applicable in a purely practical scenario, since any kind of practical algorithm constructing path decompositions can be used to compute marking sequences (the additional tasks of building G_α and the translation of a path decomposition for it back to a marking sequence are computationally simple). This observation is particularly interesting since developing practical algorithms constructing tree and path decompositions of small width is a vibrant research area. See, e.g., the work [14] and the references therein for practical algorithms constructing path decompositions; also note that designing exact and heuristic algorithms for constructing tree decompositions was part of the "PACE 2017 Parameterized Algorithms and Computational Experiments Challenge" [17].

As mentioned several times already, our reductions to and from the problem of computing the locality number also establish the locality number for words as a (somewhat unexpected) link between the graph parameters cutwidth and pathwidth. We shall discuss in more detail in Section 6 the consequences of this connection. Next, we conclude this section by providing a formal proof of Lemma 5.7, which is the main result of this section.

5.3 Proof of Lemma 5.7

In order to prove Lemma 5.7, we shall prove the two claims $\mathsf{pw}(G_\alpha) \leq 2\,\mathsf{loc}(\alpha)$ and $\mathsf{loc}(\alpha) \leq \mathsf{pw}(G_\alpha)$ separately. Recall that for any word α, by G_α we denote the graph constructed as described in Section 5.2.

We first prove $\mathsf{pw}(G_\alpha) \leq 2\,\mathsf{loc}(\alpha)$. Intuitively speaking, we will translate the stages of a marking sequence σ for α into steps of a pd-marking scheme for G_α in a natural way: each marked block $\alpha[s..t]$ is represented by letting the border positions s and t be `active`, the internal position $s+1, s+2, \ldots, t-1$ `closed`, and all other positions `open`. In particular, this means that each stage of the marking sequence with k marked blocks is represented by at most $2k$ `active` positions in the corresponding step of the pd-marking scheme (note that marked blocks of size 1 are represented by only one `active` position). The difficulty will be to show that in the process of transforming one such step of the pd-marking scheme into the next one, we do not produce more than $2\pi_\sigma(\alpha) + 1$ `active` positions. This is non-trivial, since due to the cover-property of the pd-marking scheme,

we must first set *all* positions to `active` that correspond to occurrences of the next symbol to be marked by σ before we can set them from `active` to `closed`.

Lemma 5.11. *Let α be a word. Then $\mathsf{pw}(G_\alpha) \leq 2\,\mathsf{loc}(\alpha)$.*

Proof. We first observe that there is a natural correspondence between any marked version of α and a step of a marking scheme of G_α (recall the terminology introduced in the last paragraph of Section 2.3). More precisely, every marked block $\alpha[s..t]$ can be represented by having the *border positions* s and t of G_α marked as `active`, and all *internal positions* j with $s+1 \leq j \leq t-1$ marked as `closed`. All other positions that are unmarked in α are `open` in G_α. Note that $s = t$ means that the marked block is represented by only one `active` position and no `closed` positions, and that $t = s+1$ means that the marked block is represented by two `active` positions and no `closed` positions. Hence, each of α's marked blocks of size 1 is represented by only one `active` position, while each marked block of size at least 2 is represented by two `active` positions.

In this way, a marking sequence $\sigma = (x_1, x_2, \ldots, x_m)$ for α with $\pi_\sigma(\alpha) = k$ translates into steps $p_1, p_2, \ldots, p_m$ (i. e., p_i represents stage i of σ as described above) of a marking scheme for G_α. By our observation from above, it is obvious that at each step p_i there are at most $2k_i$ `active` vertices, where k_i is the number of marked blocks at stage i of σ. It now remains to show how the steps $p_1, p_2, \ldots, p_m$ representing σ's stages can be transformed into a pd-marking scheme of G_α. More precisely, we have to describe how we can obtain step p_{i+1} from step p_i, how we can reach step p_1 from Q's initial step (i. e., where all positions are `open`), and how to transform step p_m into the final step of Q where all positions are `closed`. Moreover, this should be done in such a way that the marking scheme is a pd-marking scheme, and such that at most $2k+1$ positions are `active` in each step.

We can reach step p_1 from Q's initial step by just setting all positions of $\mathsf{ps}_{x_1}(\alpha)$ to `active` (note that $|\,\mathsf{ps}_{x_1}(\alpha)| = k_1 \leq k$), and the final step of Q can be obtained from step p_m by setting the only `active` positions 1 and $|\alpha|$ to `closed` (note that at stage m of σ the whole word is one marked block, so, by definition of step p_m, only positions 1 and $|\alpha|$ are `active`, while all other positions are `closed`). Obviously, the maximum number of `active` positions for these steps is bounded by $\max\{k_1, 2\} \leq k$.

Let s be arbitrary with $1 \leq s \leq m-1$. We now describe how step p_{s+1} can be obtained from step p_s. Intuitively speaking, we first set all positions from $\mathsf{ps}_{x_{s+1}}(\alpha)$ that extend already marked blocks to `active` (note that this includes positions that join two already marked blocks). We have to set each of these positions to `active` one after the other, and whenever some `active` position becomes an internal position of a marked block, then it must be set to `closed` so that we do not get too many `active` positions. However, we only set internal `active` positions to `closed` if they are not from $\mathsf{ps}_{x_{s+1}}(\alpha)$, since due to the fact that $\mathsf{ps}_{x_{s+1}}(\alpha)$ is a clique in G_α, we must reach a point where all positions from $\mathsf{ps}_{x_{s+1}}(\alpha)$ are `active` at the same time. After this is done, we mark the remaining positions $\mathsf{ps}_{x_{s+1}}(\alpha)$ that create new marked blocks of size 1. Let us now formally describe this marking scheme.

We call every $j \in \mathsf{ps}_{x_{s+1}}(\alpha)$ *extending*, if marking position j extends an already marked block, and we call it *isolated*, if marking position j creates a new marked block of size 1. First, we set all extending positions $j \in \mathsf{ps}_{x_{s+1}}(\alpha)$ to `active` (in some order), but every time we do this, we perform the following update operation before setting the next position to `active`: every `active` position from $\{1, 2, \ldots, |\alpha|\} \setminus \mathsf{ps}_{x_{s+1}}(\alpha)$ that is an internal position of some marked block is set to `closed`. As soon as all extending positions are `active`, we set all isolated $j \in \mathsf{ps}_{x_{s+1}}(\alpha)$ to `active`.

We have now reached the following situation (which we denote by step p'_s):

- All positions of $\mathsf{ps}_{x_{s+1}}(\alpha)$ are `active`.
- All border positions of marked blocks of stage $s+1$ of σ are `active`.
- All internal positions of marked blocks of stage $s+1$ of σ are `closed`, except possibly some of the positions from $\mathsf{ps}_{x_{s+1}}(\alpha)$ that have now become internal positions of marked blocks.

We can now transform step p'_s into step p_{s+1} by setting all `active` positions from $\mathsf{ps}_{x_{s+1}}(\alpha)$ to `closed` that are internal positions of marked blocks. This only decreases the number of `active` positions.

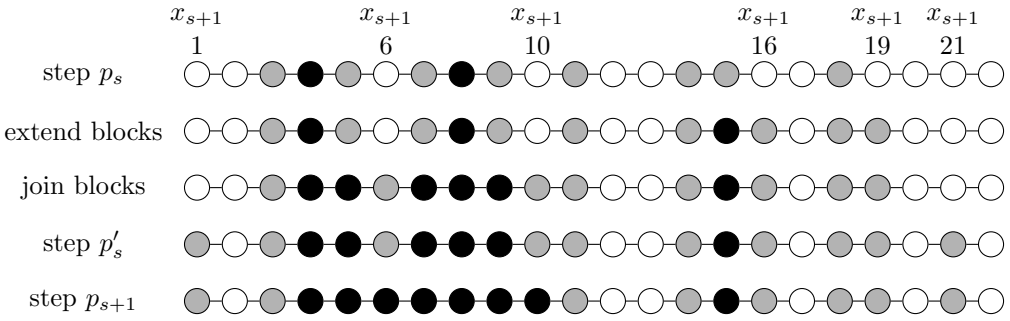

Figure 7: An illustration of how step p_s is transformed into step p_{s+1} (for simplicity, the edges that connect the sets $\mathtt{ps}_{x_i}(\alpha)$ into cliques are omitted). As in Figure 3, white vertices are `open`, grey vertices are `active`, and black vertices are `closed`. The upper most graph represents step p_s. As indicated above, we have $\mathtt{ps}_{x_i}(\alpha) = \{1, 6, 10, 16, 19, 21\}$; positions $6, 10, 16, 19$ are extending and $1, 21$ are isolated. If we first set the extending positions to `active` that do not join marked blocks, i. e., positions 16 and 19, then we obtain the situation represented by the second graph. Note that after setting 16 to `active`, we have to immediately set 15 to `closed`, whereas position 19 does not trigger such an action. Next, we set all remaining extending positions 6 and 10 to `active`, which yields the third graph. Immediately after setting 6 to `active`, we have to set both 5 and 7 to `closed`, and immediately after setting 10 to `active`, we have to set 9 to `closed`. Now it only remains to set the isolated positions 1 and 21 to `active`, which yields the second to last graph corresponding to step p'_s. Finally, in order to reach step p_{s+1}, we set all `active` positions from $\mathtt{ps}_{x_i}(\alpha)$ to `closed` that are internal positions, which sets 6 and 10 to `closed`.

This completes the definition of the marking scheme. Figure 7 contains an example of how step p_{s+1} is obtained from step p_s. In this example, we first set extending positions to `active` that do not join marked blocks, and then we set the remaining extending positions to `active`. This is done for illustrational reasons (recall that we have not restricted the order in which we set extending positions to `active`).

Next, we observe that this marking scheme is a pd-marking scheme. To this end, we observe that every edge $\{j, j+1\}$ with $1 \leq j \leq |\alpha| - 1$ is covered, since for every edge $\{j, j+1\}$, we either set j from `open` to `active` while $j+1$ is already `active`, or the other way around. Moreover, all positions from $\mathtt{ps}_{x_j}(\alpha)$ are `active` at step p'_j; thus, the cover property is also satisfied with respect to these edges.

Finally, we have to show that in this pd-marking scheme, the maximum number of `active` positions is bounded by $2k + 1$. This is obviously true at step p_1. Now let s with $1 \leq s \leq |\alpha| - 1$ be arbitrary. Since the total number of `active` positions at step p_s and p_{s+1} are bounded by $2k$, we only have to show that the maximum number of `active` positions in the marking scheme transforming p_s into p_{s+1} is bounded by $2k + 1$. Let us assume that at stage s and $s + 1$ of σ, there are k_s (k_{s+1}, respectively) marked blocks, and exactly $k_{s,1}$ ($k_{s+1,1}$, respectively) blocks have size 1; note that this means that at step p_s there are $k_{s,1} + 2(k_s - k_{s,1})$ `active` positions.

In the first phase of the marking scheme, i. e., the phase where we only set extending positions to `active`, the following different situations can arise, whenever we set some position j to `active` (see Figure 7 for an illustration):

1. j extends a marked block of size 1, but does not join two blocks: The number of `active` positions increases by 1.
   This is due to the fact that by setting j to `active`, we do not create any internal `active` positions that could be set to `closed`.

2. j extends a marked block of size at least 2, but does not join two blocks: The number of `active` positions increases by 1 and then decreases by 1.
   Assume that the block of size 2 is extended to the right. Then $j - 1$ must be `active` and,

since the block has size at least 2, $j-2$ cannot be `open`. Moreover, since $j-1$ is a neighbour of j it cannot be an element from $\mathtt{ps}_{x_{s+1}}(\alpha)$. This means that $j-1$ is set to `closed`.

3. j joins two marked blocks of size 2: the number of `active` positions increases by 1 and then decreases by 2.
   We can argue similarly as in the previous case. Positions $j-1$ and $j+1$ must be `active` and, since the blocks have size at least 2, neither $j-2$ nor $j+2$ can be `open`. Moreover, since both $j-1$ and $j+1$ are neighbours of j they cannot be elements from $\mathtt{ps}_{x_{s+1}}(\alpha)$. This means that $j-1$ and $j+1$ are set to `closed`.

4. j joins a marked block of size 1 and a block of size 2: the number of `active` positions increases by 1 and then decreases by 1.
   Without loss of generality, assume that the block of size 2 is to the left of the block of size 1. Then $j-1$ must be `active` and, since the block has size at least 2, $j-2$ cannot be `open`. Moreover, since $j-1$ is a neighbour of j it cannot be an element from $\mathtt{ps}_{x_{s+1}}(\alpha)$. This means that $j-1$ is set to `closed`.

5. j joins two blocks of size 1: the number of `active` positions increases by 1.
   This is due to the fact that by setting j to `active`, we do not create any internal `active` positions that could be set to `closed`.

We note that only operations of Type 1 and 5 increase the overall number of `active` positions. In the worst case, we apply all these operations first, before performing the other operations that potentially decrease the number of `active` positions. Let us define ℓ_s to be the number of `active` positions at step p_s, and $k_{s,1}$ to be the number of marked blocks of size 1 at stage s. Since any original marked block of size 1 can be responsible for at most one operation of Type 1 or 5 (after such an operation, the block is not of size 1 anymore), the maximum number of `active` positions after the first phase of the marking is at most $\ell_s + k_{s,1} + 1$ (we have to count "+1" since also in the operations that do not increase the overall number of `active` positions, we always have to first set a position to `active` and then, in a new step of the marking scheme, we can set another position to `closed`). Since at step p_s every marked block of size at least 2 is represented by two `active` positions, and every marked block of size 1 is represented by only one `active` position, we have $\ell_s = 2(k_s - k_{s,1}) + k_{s,1} = 2k_s - k_{s,1}$, and therefore $\ell_s + k_{s,1} + 1 = 2k_s - k_{s,1} + k_{s,1} + 1 = 2k_s + 1 \leq 2k+1$. This means that the number of `active` positions is bounded by $2k+1$ until we reach the situation where we first set an isolated position to `active`.

When we start setting isolated positions to `active`, we increase the number of `active` positions until we have reached step p'_s. Hence, we have to bound the total number of `active` positions at step p'_s.

For the following reasoning, let us assume that in going from stage s to stage $s+1$ in σ, we mark all occurrences of x_{s+1} one after the other (instead of all of them in parallel). Each of these individual markings can either create a new marked block of size 1, or join an existing marked block with another existing marked block, or just extend a marked block (possibly of size 1) without joining any marked blocks. Let us assume that creating a marked block of size 1 happens q times, and joining two marked blocks happens t times (how often we extend marked blocks without joining is not important).

We first count the number of `active` positions at step p'_s that correspond to border positions of marked blocks. For each marked block of size at least 2 there are 2 such `active` positions, while for each block of size 1 there is only 1 such `active` position. Consequently, there are $2(k_{s+1} - k_{s+1,1}) + k_{s+1,1} = 2k_{s+1} - k_{s+1,1}$ such border positions, where $k_{s+1,1}$ is the number of marked blocks of size 1. Since $q \leq k_{s+1,1}$, we have $2k_{s+1} - k_{s+1,1} \leq 2k_{s+1} - q$ border positions. In addition to these border positions, we also have a number of `active` positions that are internal positions of marked blocks. However, each such internal `active` position results from joining two blocks, which means that we have r such positions. Hence, we have at most $2k_{s+1} - q + r$ `active` positions. Since stage $s+1$ of σ is obtained from stage s by creating q new blocks and joining r marked blocks (and extending some blocks), we have $k_{s+1} = k_s + q - r$. Consequently, $2k_{s+1} - q + r = k_{s+1} + k_s + q - r - q + r = k_{s+1} + k_s \leq 2k$. This means that the maximum number of `active` positions in the marking scheme that transforms step p_s into step p_{s+1} is bounded by $2k+1$.

Consequently, we have described a pd-marking scheme for G_α that has always at most $2k+1$ `active` positions, which means that $\mathsf{pw}(G_\alpha) \leq 2k$. □

Next, we take care of the other inequality $\mathsf{loc}(\alpha) \leq \mathsf{pw}(G_\alpha)$. On an intuitive level, the proof will proceed as follows. Any pd-marking scheme Q for G_α induces a linear order on $\{x_1, x_2, \ldots, x_m\}$ (and therefore a marking sequence σ), since it is forced to go through individual steps where all positions of the cliques $\mathsf{ps}_{x_i}(\alpha)$ with $1 \leq i \leq m$ are `active` at the same time. It is our goal to prove that there are at least $\pi_\sigma(\alpha)+1$ fl position in the pd-marking scheme Q, since this implies that $\mathsf{loc}(\alpha) \leq \pi_\sigma(\alpha) \leq \mathsf{w}(Q)$.

The marking sequence q has a stage s in which the maximum of $\pi_\sigma(\alpha)$ marked blocks is reached for the first time. In the corresponding step p_s of the pd-marking scheme (i. e., the step where the positions of $\pi_\sigma(\alpha)$ are all `active` for the first time), we obviously cannot assume that the marked blocks are represented in the way of the proof of Lemma 5.11 (i. e., border positions are `active`, internal positions are `closed`, and all other positions are `open`). However, by carefully analysing step p_s, we can identify at least $\pi_\sigma(\alpha)$ `active` positions (for this, we need the property that Q is a pd-marking scheme for G_α and that a maximum number of marked blocks is reached at stage s of σ). If now there is one additional `active` position, then there are $\pi_\sigma(\alpha)+1$ `active` positions and we are done. This is the easy part of the proof, and the the more difficult part is the case where we do not have an additional `active` position, i. e., the identified $\pi_\sigma(\alpha)$ `active` positions are the only `active` positions. This property, however, can be shown to impose some structural constraints with respect to step p_s of the pd-marking sequence and stage s of the marking sequence. In particular, by some more technical combinatorial observations and exhaustive case distinctions, we are able to prove that we will necessarily get $\pi_\sigma(\alpha)+1$ `active` positions in the next step of the pd-marking sequence, or we can prove that the marking sequence σ can be changed into a better marking sequence σ' with $\pi_{\sigma'}(\alpha) = \pi_\sigma(\alpha) - 1$. This latter property means that we have $\mathsf{loc}(\alpha) \leq \pi_\sigma(\alpha) - 1 \leq \mathsf{w}(Q)$.

Lemma 5.12. *Let α be a word with $|\alpha| \geq 2$. Then, given any path-decomposition Q for G_α, a marking sequence σ for α with $\mathsf{loc}(\sigma) \leq \mathsf{w}(Q)$ can be constructed in $\mathrm{O}(|\alpha|)$. In particular, $\mathsf{loc}(\alpha) \leq \mathsf{pw}(G_\alpha)$.*

Proof. Let $Q = (B_0, B_1, B_2, \ldots, B_{2|\alpha|})$ be an arbitrary nice path-decomposition for G_α, which we interpret as a pd-marking scheme (see the last paragraph of Section 2.3). For every i, $1 \leq i \leq m$, $\mathsf{ps}_{x_i}(\alpha)$ is a clique in G_α; thus, there must be a step p_i of this pd-marking sequence in which all positions of $\mathsf{ps}_{x_i}(\alpha)$ are `active` for the first time. Without loss of generality, we assume that $p_1 < p_2 < \ldots < p_m$. Next, let $\sigma = (x_1, x_2, \ldots, x_m)$ be the marking sequence for α that marks the symbols $x_1, x_2, \ldots, x_m$ in the same order as their occurrences $\mathsf{ps}_{x_i}(\alpha)$ are all together `active` for the first time in the pd-marking scheme. Thus σ can be constructed from Q in time $\mathrm{O}(|\alpha|)$. We now prove that for $k = \pi_\sigma(\alpha)$ one of the following cases holds:

- There is a step of Q with at least $k+1$ `active` positions. In this case, we have $\mathsf{loc}(\alpha) \leq k \leq \mathsf{w}(Q)$.
- There is a step of Q with at least k `active` positions and a marking sequence σ' with $\pi_{\sigma'}(\alpha) = k-1$. In this case, we have $\mathsf{loc}(\alpha) \leq k-1 = \mathsf{w}(Q)$.

Since the path decomposition is arbitrary, this implies that $\mathsf{loc}(\alpha) \leq \mathsf{w}(Q)$ for every path decomposition Q, and therefore also $\mathsf{loc}(\alpha) \leq \mathsf{pw}(G_\alpha)$.

Let s, $1 \leq s \leq m$, be chosen such that the maximum number of marked blocks is reached for the first time at stage s of σ, i. e., after marking symbol x_s, we obtain k marked blocks for the first time. As defined above, p_s is the step of Q where all positions of $\mathsf{ps}_{x_s}(\alpha)$ are `active` for the first time. We now represent the pd-marking scheme at step p_s and the marked version of α at stage s of σ as a single marked word $\widehat{\alpha}$ over the alphabet $\{\mathtt{o}, \mathtt{a}, \mathtt{c}\}$. More precisely, for every i with $1 \leq i \leq |\alpha|$, $\widehat{\alpha}[i] = \mathtt{o}$ if position i is `open` at step p_s, $\widehat{\alpha}[i] = \mathtt{a}$ if position i is `active` at step p_s and $\widehat{\alpha}[i] = \mathtt{c}$ if position i is `closed` at step p_s of the pd-marking scheme. This fully describes step p_s of the pd-marking scheme. Moreover, we represent the marked version of α at stage s of σ by marking the symbols of $\widehat{\alpha}$ in the same way, i. e., for every i with $1 \leq i \leq |\alpha|$, the symbol $\widehat{\alpha}[i]$ is marked if $\alpha[i]$ is marked at stage s of σ (i. e., $\alpha[i] \in \{x_1, x_2, \ldots, x_s\}$), and $\widehat{\alpha}[i]$ is unmarked

otherwise. We shall also consider $\widehat{\alpha}$'s factorisation according to its marked and unmarked blocks, i. e., we consider the factorisation

$$\widehat{\alpha} = \beta_0\mu_1\beta_1\mu_2\ldots\mu_k\beta_k\,,$$

where the factors β_i, $0 \leq i \leq k$, correspond to the unmarked blocks of $\widehat{\alpha}$, and μ_i, $1 \leq i \leq k$, correspond to the marked blocks of $\widehat{\alpha}$. Next, we establish some simple properties of $\widehat{\alpha}$ and its factorisation.

1. If $\widehat{\alpha}[i] = \mathtt{c}$ for some i, $1 \leq i \leq |\alpha|$, then position i is `closed` at step p_s of the pd-marking scheme, which means that the situation where all positions of $\mathtt{ps}_{\alpha[i]}(\alpha)$ have been `active` at the same time must have occurred already. This means that $\alpha[i] \in \{x_1, x_2, \ldots, x_s\}$. Hence, position i is marked and therefore in some marked block. Consequently, all of $\widehat{\alpha}$'s occurrences of `c` occur in marked blocks μ_j, i. e., $\beta_0, \beta_k \in \{\mathtt{a}, \mathtt{o}\}^*$, $\beta_i \in \{\mathtt{a}, \mathtt{o}\}^+$ and $\mu_i \in \{\mathtt{a}, \mathtt{c}\}^+$ for every i with $1 \leq i \leq k$ (note that the factors μ_i cannot contain occurrences of `o` by definition of σ (i. e., an x_i is marked by σ not before all positions $\mathtt{ps}_{x_i}(\alpha)$ are `active`)).

2. For every i, $1 \leq i \leq k-1$, if the last symbol of the marked block μ_i is `c`, then the first symbol of the unmarked block β_i must be `a` (or $\beta_i = \varepsilon$, which can happen for $i = k$), since otherwise β_i's first symbol must be `o` (see Point 1), which leads to the contradiction that at step p_s of the pd-marking scheme there is a `closed` position adjacent to an `open` position (this violates the cover property of path decompositions). Analogously, it follows that if the first symbol of the marked block μ_i is `c`, then the first symbol of the unmarked block β_{i-1} must be `a` (or $\beta_{i-1} = \varepsilon$, which can happen for $i = 1$).

3. For every i with $1 \leq i \leq |\alpha|$ such that $\alpha[i] = x_s$, position i must be contained in some marked block of $\widehat{\alpha}$ (since x_s is marked at stage s of σ), and (by definition of σ) position i must be `active` at step p_s of the pd-marking scheme, i. e., $\widehat{\alpha}[i] = \mathtt{a}$. Moreover, there must be at least one such i with $1 \leq i \leq |\alpha|$ such that $\alpha[i] = x_s$.

With Point 3, there is some j with $1 \leq j \leq |\alpha|$ such that $\alpha[j] = x_s$, and some r with $1 \leq r \leq k$ such that j is a position of the marked block μ_r of $\widehat{\alpha}$. Next, for every i with $1 \leq i \leq k$, we define a position t_i with $\widehat{\alpha}[t_i] = \mathtt{a}$ that either lies in t_i, or is the first position of β_i, or the last position of β_{i-1}. First, we set $t_r = j$. For every i, $1 \leq i < r$, we let t_i be some position of μ_i that is an occurrence of `a` if one exists. If, on the other hand, μ_i has no occurrences of `a`, then, due to Point 1, μ_i's last symbol is `c`, and, by Point 2, this means that β_i's first symbol is `a`, so we let t_i be β_i's first position. Analogously, for every i with $r < i \leq k$, we let t_i be some position of μ_i that is an occurrence of `a` if one exists, and if μ_i has no occurrences of `a`, then its first symbol is `c`, which means that β_{i-1}'s last symbol is `a`, so we let t_i be β_{i-1}'s last position.

Since every t_i with $1 \leq i < r$ is in $\mu_i\beta_i[1]$, every t_i with $r < i \leq k$ is in $\beta_{i-1}[|\beta_i|]\mu_i$, and t_r is in μ_r, these positions t_i are in fact k distinct positions that are `active` at step p_s of the pd-marking scheme.

Now, if there is at least one additional `active` position, then there are at least $k+1$ `active` positions at step p_s, i. e., we have arrived at the first of the two cases mentioned above. In order to conclude the proof, we assume now that the `active` positions t_i, $1 \leq i \leq k$, are the only `active` positions at step p_s of the pd-marking scheme, and we will show that this either leads again to the first case, but with respect to some other step of the pd-marking scheme, or to the second case mentioned above, i. e., there is a marking sequence σ' with $\pi_{\sigma'}(\alpha) = k-1$.

First, we divide $\widehat{\alpha}$ into the part left of μ_r, the factor μ_r and the part right of μ_r, i. e., we consider the factorisation $\widehat{\alpha} = \widehat{\alpha}_1\mu_r\widehat{\alpha}_2$, where we call $\widehat{\alpha}_1 = \beta_0\mu_1\beta_1\mu_2\ldots\beta_{r-1}$ the *left side* and we call $\widehat{\alpha}_2 = \beta_r\mu_{r+1}\beta_{r+1}\ldots\mu_k\beta_k$ the *right side*. By our assumption that the positions t_i with $1 \leq i \leq k$ are the only occurrences of `a` in $\widehat{\alpha}$, and the Points 1 to 3 from above, we can conclude several facts about the form of the left and the right side. In the following, we shall only analyse the left side; all the following arguments apply analogously to the right side as well.

Each position t_ℓ with $1 \leq \ell < r$ is either inside μ_ℓ, or it is the leftmost position of β_ℓ. If it is the leftmost position of β_ℓ, then there is no occurrence of `a` in μ_ℓ, which means that the leftmost symbol in μ_ℓ must be `c`, and therefore, the rightmost symbol of $\beta_{\ell-1}$ must be an `a` (note that $\beta_{\ell-1} \in \{\mathtt{a}, \mathtt{o}\}^+$ (see Point 1) and it is not possible that a symbols `o` occurs next to a symbol

$\mathtt{c}$). However, this rightmost position must be the position $t_{\ell-1}$ and is therefore also the leftmost position of $\beta_{\ell-1}$. In particular, this means that $\beta_{\ell-1} = \mathtt{a}$ and $\mu_{\ell-1} \in \{\mathtt{c}\}^+$. This inductively proceeds to the left and therefore also means that $\beta_0 = \varepsilon$. Therefore, if ℓ with $1 \leq \ell < r$ is maximal such that t_ℓ is not inside μ_ℓ, then we have the following situation:

$$\widehat{\alpha}_1 = \mu_1 \underbrace{\mathtt{a}}_{\beta_1} \mu_2 \underbrace{\mathtt{a}}_{\beta_2} \dots \mu_{\ell-1} \underbrace{\mathtt{a}}_{\beta_{\ell-1}} \mu_\ell \underbrace{\mathtt{a}\,\mathtt{o}^{g_1}}_{\beta_\ell} \mu_{\ell+1} \beta_{\ell+1} \mu_{\ell+2} \dots \beta_{r-2} \mu_{r-1} \beta_{r-1} \,,$$

where $\mu_i \in \{\mathtt{c}\}^+$ for every i with $1 \leq i \leq \ell$, and $g_1 \geq 0$. Moreover, since ℓ is maximal, all μ_i with $\ell + 1 \leq i \leq r-1$ contain an `active` position, which means that $\beta_i \in \{\mathtt{o}\}^+$ for every i with $\ell+1 \leq i \leq r-1$. However, this directly implies that $\mu_i = \mathtt{a}$ with $\ell+2 \leq i \leq r-1$ and $\mu_{\ell+1} = \mathtt{c}^{g_2}\,\mathtt{a}$ for some $g_2 \geq 0$ with the property that at most one of g_1 and g_2 can be positive. More precisely, we have the following situation:

$$\widehat{\alpha}_1 = \mu_1 \underbrace{\mathtt{a}}_{\beta_1} \mu_2 \underbrace{\mathtt{a}}_{\beta_2} \dots \mu_{\ell-1} \underbrace{\mathtt{a}}_{\beta_{\ell-1}} \mu_\ell \underbrace{\mathtt{a}\,\mathtt{o}^{g_1}}_{\beta_\ell} \underbrace{\mathtt{c}^{g_2}\,\mathtt{a}}_{\mu_{\ell+1}} \beta_{\ell+1} \underbrace{\mathtt{a}}_{\mu_{\ell+2}} \dots \beta_{r-2} \underbrace{\mathtt{a}}_{\mu_{r-1}} \beta_{r-1} \,, \tag{$\dagger$}$$

where $\beta_i \in \{\mathtt{o}\}^+$ for every i with $\ell + 1 \leq i \leq r-1$, $\mu_i \in \{\mathtt{c}\}^+$ for every i with $1 \leq i \leq \ell$, and $g_1, g_2 \geq 0$ with $0 \in \{g_1, g_2\}$.

If, on the other hand, no such ℓ, $1 \leq \ell < r$, exists, then all the `active` positions t_i are in μ_i for every i with $1 \leq i \leq r-1$. In particular, this means that $\beta_i \in \{\mathtt{o}\}^+$ for every i with $1 \leq i \leq r-1$, which forces all μ_i, $2 \leq i \leq r-1$, to start and end with an `active` position, while μ_1 must have a rightmost `active` position, but a leftmost `active` position only if $\beta_0 \neq \varepsilon$. Thus,

$$\widehat{\alpha}_1 = \beta_0 \underbrace{\mathtt{c}^g\,\mathtt{a}}_{\mu_1} \beta_1 \underbrace{\mathtt{a}}_{\mu_2} \dots \beta_{r-2} \underbrace{\mathtt{a}}_{\mu_{r-1}} \beta_{r-1} \,, \tag{$\star$}$$

where $\beta_i \in \{\mathtt{o}\}^+$ for every i with $1 \leq i \leq r-1$, $g \geq 0$, and $g > 0$ implies $\beta_0 = \varepsilon$.

As mentioned above, these observations also apply to the right side $\widehat{\alpha}_2$ in an analogous way.

Next, we turn to the marked block μ_r in between the left and right side of $\widehat{\alpha}$. Since μ_r contains exactly one occurrence of $\mathtt{a}$, we can factorise it into $\mu_r = \nu_1\,\mathtt{a}\,\nu_2$, where $\nu_1, \nu_2 \in \{\mathtt{c}\}^*$. We now consider four individual cases that arise from whether or not ν_1 or ν_2 are empty. For each of these cases, we can use the observations from above in order to further determine the structure of the left or right side of $\widehat{\alpha}$.

Claim (1) If $\nu_1 \neq \varepsilon$, then we have

$$\widehat{\alpha}_1 = \mu_1\,\mathtt{a}\,\mu_2\,\mathtt{a} \dots \mu_{r-1}\,\mathtt{a}\,.$$

Proof: If $\nu_1 \neq \varepsilon$, then $\nu_1[1] = \mathtt{c}$, which implies that $\beta_{r-1}[|\beta_{r-1}|] = \mathtt{a}$ and therefore $\beta_{r-1} = \mathtt{a}$. This means that for $\ell = r-1$, we have the case described in ($\dagger$), i. e., where t_ℓ is the rightmost `active` position that is not in μ_ℓ and, since $\beta_\ell = \mathtt{a}$, we also have the case $g_1 = 0$. This directly implies the statement claimed above. □

Claim (2) If $\nu_2 \neq \varepsilon$, then we have

$$\widehat{\alpha}_2 = \mathtt{a}\,\mu_{r+1}\,\mathtt{a}\,\mu_{r+2} \dots \mathtt{a}\,\mu_k\,.$$

Proof: Analogous to **Claim (1)**. □

Claim (3) If $\nu_1 = \varepsilon$, then we have one of the following two cases:

(a) For some ℓ with $1 \leq \ell \leq r-1$,

$$\widehat{\alpha}_1 = \mu_1 \underbrace{\mathtt{a}}_{\beta_1} \mu_2 \underbrace{\mathtt{a}}_{\beta_2} \dots \mu_{\ell-1} \underbrace{\mathtt{a}}_{\beta_{\ell-1}} \mu_\ell \underbrace{\mathtt{a}\,\mathtt{o}^{g_1}}_{\beta_\ell} \underbrace{\mathtt{c}^{g_2}\,\mathtt{a}}_{\mu_{\ell+1}} \beta_{\ell+1} \underbrace{\mathtt{a}}_{\mu_{\ell+2}} \dots \beta_{r-2} \underbrace{\mathtt{a}}_{\mu_{r-1}} \beta_{r-1} \,,$$

where $\beta_i \in \{\mathtt{o}\}^+$ for every i with $\ell + 1 \leq i \leq r-1$, $\mu_i \in \{\mathtt{c}\}^+$ for every i with $1 \leq i \leq \ell$, and $g_1, g_2 \geq 0$ with $0 \in \{g_1, g_2\}$. Note that this is exactly the case described in Equation †.

(b) $\widehat{\alpha}_1 = \beta_0 \underbrace{\mathtt{c}^g\,\mathtt{a}}_{\mu_1} \beta_1 \underbrace{\mathtt{a}}_{\mu_2} \dots \beta_{r-2} \underbrace{\mathtt{a}}_{\mu_{r-1}} \beta_{r-1}$,
where $\beta_i \in \{\mathtt{o}\}^+$ for every i with $1 \leq i \leq r-1$, $g \geq 0$, and $g > 0$ implies $\beta_0 = \varepsilon$. Note that this is exactly the case described in Equation $\star$.

Proof: If there is some ℓ', $1 \leq \ell' \leq r-1$, such that $t_{\ell'}$ is not in $\mu_{\ell'}$, then we can consider a maximal ℓ with this property and can conclude that we have the case described in (†), which is exactly the statement of **Claim (3)**a. If, on the other hand, no such ℓ' exists, then we have the case described in ($\star$), which is exactly the statement of **Claim (3)**b. □

Claim (4) If $\nu_2 = \varepsilon$, then we have one of the following two cases:

(a) For some ℓ with $r \leq \ell \leq k$,

$$\widehat{\alpha}_2 = \beta_r \underbrace{\mathtt{a}}_{\mu_{r+1}} \beta_{r+1} \underbrace{\mathtt{a}}_{\mu_{r+2}} \dots \beta_{\ell-1} \underbrace{\mathtt{a}\,\mathtt{c}^{g_1}}_{\mu_\ell} \underbrace{\mathtt{o}^{g_2}\,\mathtt{a}}_{\beta_\ell} \mu_{\ell+1} \underbrace{\mathtt{a}}_{\beta_{\ell+1}} \mu_{\ell+2} \underbrace{\mathtt{a}}_{\beta_{\ell+2}} \dots \underbrace{\mathtt{a}}_{\beta_{k-1}} \mu_k\,,$$

where $\beta_i \in \{\mathtt{o}\}^+$ for every i with $r \leq i \leq \ell-1$, $\mu_i \in \{\mathtt{c}\}^+$ for every i with $\ell+1 \leq i \leq k$, and $g_1, g_2 \geq 0$ with $0 \in \{g_1, g_2\}$.

(b) $\widehat{\alpha}_2 = \beta_r \underbrace{\mathtt{a}}_{\mu_{r+1}} \beta_{r+1} \underbrace{\mathtt{a}}_{\mu_{r+2}} \dots \underbrace{\mathtt{a}\,\mathtt{c}^g}_{\mu_1} \beta_k$,
where $\beta_i \in \{\mathtt{o}\}^+$ for every i with $r \leq i \leq k-1$, $g \geq 0$, and $g > 0$ implies $\beta_k = \varepsilon$.

Proof: Analogous to **Claim (3)**. □

In order to conclude the proof, we need some preliminary observations and definitions.

> **Observation (∗)**: Let i be arbitrary with $1 \leq i \leq |\alpha|$. Position i is marked at stage s of σ (and therefore in some marked block of $\widehat{\alpha}$) if and only if $\alpha[i] \in \{x_1, x_2, \dots, x_s\}$. If $\alpha[i] = x_s$, then position i must be `active` at step p_s (and therefore $\widehat{\alpha}[i] = \mathtt{a}$). If $\alpha[i] \in \{x_1, x_2, \dots, x_{s-1}\}$, then position i must be `active` or `closed` at step p_s (and therefore $\widehat{\alpha}[i] \in \{\mathtt{a}, \mathtt{c}\}$).

Let i with $1 \leq i \leq |\alpha|$ be some position that is `active` at step p_s of the pd-marking scheme. We say that i is *blocked* if it is unmarked in $\widehat{\alpha}$ or if $\mathtt{o} \in \{\widehat{\alpha}[i-1], \widehat{\alpha}[i+1]\}$. The idea of this definition is that if i is blocked, then it cannot be set from `active` to `closed` in the next step of the pd-marking scheme. Indeed, if i is not marked, then $\alpha[i] \in \{x_{s+1}, x_{s+2}, \dots, x_m\}$ (see Observation (∗)), and, by assumption, for symbols $x \in \{x_{s+1}, x_{s+2}, \dots, x_m\}$ we have not yet reached the situation that all positions from $\mathtt{ps}_x(\alpha)$ are `active` at the same time); thus, none of the corresponding positions can be set to `closed` in the next step. Moreover, if $\mathtt{o} \in \{\widehat{\alpha}[i-1], \widehat{\alpha}[i+1]\}$, then the `active` position i is adjacent to an `open` position and therefore cannot be set to `closed` in the next step.

We next show the following claim:

> **Claim (∗∗)**: Every position on the left side and on the right side that is `active` at step p_s of the pd-marking scheme is blocked.

Proof: Due to **Claim (1)** to **Claim (4)**, we know that every position i that is `active` in step p_s of the pd-marking scheme and corresponds to a marked occurrence of `a` on the left side, satisfies $\widehat{\alpha}[i+1] = \mathtt{o}$. Moreover, every position i that is `active` in step p_s of the pd-marking scheme and corresponds to a marked occurrence of `a` on the right side, satisfies $\widehat{\alpha}[i-1] = \mathtt{o}$. Consequently, every position on the left and right side that is `active` at step p_s of the pd-marking scheme is blocked. □

We are now ready to finally conclude this proof. To this end, we analyse all possible cases of how μ_r may look like, and we show that for each case we either obtain a contradiction, or there is a step of Q with at least $k+1$ `active` positions, or there is a marking sequence σ' with $\pi_{\sigma'}(\alpha) = k-1$ (see the two cases mentioned at the beginning of this proof).

First, we note that if t_r is blocked, then, due to Claim (∗∗), every position that is `active` at step p_s of the pd-marking scheme is blocked. Thus, an `open` position will be set to `active` in the

next step of the pd-marking scheme, which means that there are $k+1$ `active` positions in the next step of the pd-marking scheme.

If t_r is not blocked, then, since t_r is marked in $\widehat{\alpha}$, we must have $\widehat{\alpha}[t_r-1] \neq \mathtt{o}$ and $\widehat{\alpha}[t_r+1] \neq \mathtt{o}$. We consider four cases that arise from whether ν_1 or ν_2 are empty.

- $\nu_1 = \varepsilon$ and $\nu_2 \neq \varepsilon$: Since $\nu_1 = \varepsilon$, position t_r is a right neighbour of β_{r-1}. If the last symbol of β_{r-1} is an occurrence of $\mathtt{o}$, then t_r is blocked, which is a contradiction to our assumption that t_r is not blocked. Therefore, the last symbol of β_{r-1} is an occurrence of $\mathtt{a}$, which, according to **Claim (3)**, is only possible if we have the situation described in **Claim (3)**a with $\ell = r-1$ and $g_1 = 0$. Indeed, if **Claim (3)**b applies or if **Claim (3)**a applies with $\ell < r-1$, then $\beta_{r-1} \in \{\mathtt{o}\}^+$, and if **Claim (3)**a applies with $g_1 > 0$, then $\beta_{r-1} = \mathtt{a}\,\mathtt{o}^{g_1}$.

  Claim (2) implies that there is no marked position in the right side of $\widehat{\alpha}$ that is also `active` at step p_s of the pd-marking scheme. Furthermore, since we have **Claim (3)**a with $\ell = r-1$, there is also no marked position in the left side of $\widehat{\alpha}$ that is also `active` at step p_s of the pd-marking scheme. Consequently, Observation $(*)$ implies that t_r is the only position with $\alpha[i] = x_s$. Since x_s is marked in stage s of σ, ν_2 corresponds to a non-empty factor of α that is marked at stage $s-1$ of σ, and position $t_r - 1$ is not marked in stage $s-1$ of σ, we know that marking x_s in stage s just extends the marked block that corresponds to ν_2. Hence, at stage $s-1$ there were k marked blocks, which is a contradiction to the assumption that s is the first stage of σ where we reach the maximum of k marked blocks.

- $\nu_1 \neq \varepsilon$ and $\nu_2 = \varepsilon$: This case leads to a contradiction analogously to the previous case.

- $\nu_1 \neq \varepsilon$ and $\nu_2 \neq \varepsilon$: Due to **Claim (1)** and **Claim (2)**, there is no marked position in the right or left side of $\widehat{\alpha}$ that is also `active` at step p_s of the pd-marking scheme. Hence, Observation $(*)$ again implies that t_r is the only position with $\alpha[i] = x_s$. Since x_s is marked in stage s of σ, and since both ν_1 and ν_2 correspond to non-empty factors of α that are marked at stage $s-1$ of σ, we know that marking x_s in stage s joins two marked blocks and changes no other marked block. Hence, at stage $s-1$ there were $k+1$ marked blocks, which is a contradiction.

- $\nu_1 = \varepsilon$ and $\nu_2 = \varepsilon$: Since t_r is not blocked and $\beta_{r-1}\widehat{\alpha}[t_r]\beta_r$ is a factor of $\widehat{\alpha}$, we know that neither the last symbol of β_{r-1} nor the first symbol of β_r can be occurrences of `open`. Just like in the first and second case, **Claim (3)** and **Claim (4)** imply that this is only possible if with respect to the left side, we have the situation described in **Claim (3)**a with $\ell = r-1$ and $g_1 = 0$, and with respect to the right side, we have the situation described in **Claim (4)**a with $\ell = r$ and $g_2 = 0$. This means that we have the following situation

$$\widehat{\alpha} = \mu_1 \underbrace{\mathtt{a}}_{\beta_1} \mu_2 \underbrace{\mathtt{a}}_{\beta_2} \cdots \mu_{r-1} \underbrace{\mathtt{a}}_{\beta_{r-1}} \underbrace{\mathtt{a}}_{\mu_r} \underbrace{\mathtt{a}}_{\beta_r} \mu_{r+1} \underbrace{\mathtt{a}}_{\beta_{r+1}} \cdots \mu_{k-1} \underbrace{\mathtt{a}}_{\beta_{k-1}} \mu_k \,.$$

  Hence, t_r is the only `active` position that is marked in $\widehat{\alpha}$, which means that t_r is the only position with $\alpha[i] = x_s$. In particular, t_r is the only position that is unmarked in stage $s-1$ and marked in stage s of σ. This implies that in stage $s-1$ of σ there are exactly $k-1$ marked blocks (i. e., the factors μ_j with $1 \leq j \leq k$ and $j \neq r$) and, by our assumption that stage s is the first stage with k marked blocks, we also know that in stages $1, 2, \ldots, s-1$ the maximum number of marked blocks is $k-1$. Moreover, at stage $s-1$, every unmarked position except t_r (i. e., all the positions corresponding to the occurrences of $\mathtt{a}$, except the occurrence $\widehat{\alpha}[t_r] = \mathtt{a}$) is a neighbour of some marked block. Consequently, we can change σ into a marking sequence σ' as follows. The marking sequence σ' simulates σ up to stage $s-1$. As observed above, so far the maximum number of marked blocks is $k-1$. Then, instead of marking x_s, σ' marks all other unmarked symbols in some order. In each of the corresponding stages of the marking sequence, marking the next symbol leaves the number of marked blocks unchanged, or decreases it (this can be easily seen by consulting the factorisation illustrated above). Finally, symbol x_s is marked as the last symbol. Thus, σ' is a marking sequence for α with $\pi_{\sigma'}(\alpha) = k-1$. □

6 A New Relationship Between Pathwidth and Cutwidth

We observe that the reduction from MinCutwidth to MinLoc from Section 4.1 combined with the reduction from MinLoc to MinPathwidth from Section 5.2 gives a reduction from MinCutwidth to MinPathwidth. Moreover, this reduction is approximation preserving; thus, it carries over approximations for MinPathwidth (e. g., [21, 30]) to MinCutwidth, and yields new results for MinCutwidth.

Pathwidth and cutwidth are classical graph parameters that play an important role for graph algorithms, independent from our application for computing the locality number. Therefore, it is the main purpose of this section to translate the reduction from MinCutwidth to MinPathwidth that takes MinLoc as an intermediate step into a direct reduction from MinCutwidth to MinPathwidth. Such a reduction is of course implicitly hidden in the reductions of Sections 4.1 and 5.2, but we believe that explaining the connection in a more explicit way will be helpful for researchers that are mainly interested in the graph parameters cutwidth and pathwidth.

The relationship between cutwidth and pathwidth revealed by this direct reduction is best illustrated via a third graph parameter that we call *second order cutwidth*. To the best of our knowledge, this parameter has not explicitly been studied before.

Let $L = (v_1, v_2, \ldots, v_n)$ be a linear arrangement of a (multi)graph $G = (V, E)$. For the classical cutwidth, we consider the maximum number of edges that span over a gap between a vertex v_i and v_{i+1}. For the second order cutwidth on the other hand, we consider the maximum number of edges that span over a vertex v_i or that are adjacent to v_i, i. e., all edges $\{v_k, v_\ell\}$ with $k \leq i \leq \ell$ (note that this is equivalent to considering the maximum number of edges that span over the gap between v_{i-1} and v_i or over the gap between v_i and v_{i+1}). Let us now formally define the second order cutwidth.

For every $i \in \{1, 2, \ldots, n\}$, we define $\Gamma_{L,i} = \{\{v_k, v_\ell\} \in E \mid k \leq i \leq \ell\}$. The *second order cutwidth* of the linear arrangement L is defined by $\mathsf{cw}_2(L) = \max\{|\Gamma_{L,i}| \mid 1 \leq i \leq n\}$, and the second order cutwidth of G is defined by $\mathsf{cw}_2(G) = \min\{\mathsf{cw}_2(L) \mid L \text{ is a linear arrangement for } G\}$. Since $\Gamma_{L,i} = \mathcal{C}_L(i-1) \cup \mathcal{C}_L(i)$ for every $i \in \{1, 2, \ldots, n\}$, we can also define the second order cutwidth in terms of the sets $\mathcal{C}_L$, i. e., $\mathsf{cw}_2(L) = \max\{|\mathcal{C}_L(i-1) \cup \mathcal{C}_L(i)| \mid 1 \leq i \leq n\}$.

For example, the linear arrangement L on the top of Figure 2 has a second order cutwidth of 6, which, e. g., is witnessed by $\Gamma_{L,3}$, since $\Gamma_{L,3} = \{\{u, w\}, \{u, x\}, \{v, x\}, \{v, y\}, \{v, z\}, \{w, x\}\}$. Note that $\Gamma_{L,3} = \mathcal{C}_L(2) \cup \mathcal{C}_L(3)$. On the other hand, the linear arrangement L' on the bottom has a second order cutwidth of 4 (witnessed by $|\Gamma_{L',2}| = 4$).

To understand the relationship between this parameter and cutwidth, we first show the following.

Lemma 6.1. *Let $G = (V, E)$ be a connected graph with at least three vertices, then $\mathsf{cw}(G) + 1 \leq \mathsf{cw}_2(G) \leq 2\,\mathsf{cw}(G)$. Further, given a linear arrangement L for G, a linear arrangement L' for G with $\mathsf{cw}_2(L) - 1 \geq \mathsf{cw}(L')$ can be computed in $\mathrm{O}(|E|)$.*

Proof. Simply by definition, we see that

$$\begin{aligned}\mathsf{cw}_2(L) &= \max\{\Gamma_{L,i} \mid 1 \leq i \leq n\} = \max\{|\mathcal{C}_L(i-1) \cup \mathcal{C}_L(i)| \mid 1 \leq i \leq n\} \\ &\leq \max\{|\mathcal{C}_L(i-1)| + |\mathcal{C}_L(i)| \mid 1 \leq i \leq n\} \leq 2\,\mathsf{cw}(L)\,,\end{aligned}$$

for any linear arrangement L for G. This directly gives the second inequality.

For the first inequality, observe that by definition $\mathsf{cw}(L) \leq \mathsf{cw}_2(L)$ for any linear arrangement L. Let $L = (v_1, v_2, \ldots, v_n)$ be a linear arrangement of minimum second order cutwidth, and assume $\mathsf{cw}(L) = \mathsf{cw}_2(L)$ (otherwise $L' := L$ shows the claim). We show how to construct a linear arrangement L' of strictly smaller cutwidth than L in polynomial time, by iterative rearrangements. To this end, we define $I_{max}(L) = \{v_t \mid |\mathcal{C}_L(t)| = \mathsf{cw}_2(G)\}$ and let $I^1_{max}(L)$ be the subset of $I_{max}(L)$ of degree one vertices, i. e., $I_{max}(L) = \{v_t \in I_{max}(L) \mid |N(v_t)| = 1\}$. We now stepwise rearrange L and use I_{max} and I^1_{max} to track progress, never increasing the second order cutwidth of the arrangement. Our goal is a linear arrangement L' with $\mathsf{cw}_2(L') = \mathsf{cw}_2(G)$ where $I_{max}(L') = \emptyset$; note that this implies the desired $\mathsf{cw}(L') \leq \mathsf{cw}_2(G) - 1$. Until we have reached this goal, we have a linear arrangement L with $\mathsf{cw}(L) = \mathsf{cw}_2(L)$ implying $\mathcal{C}_L(t) = \Gamma_{L,t}$ for each $t \in I_{max}(L)$. Since $\Gamma_{L,t} = \{\{v_k, v_\ell\} \in E \mid k \leq t \leq \ell\}$ and $\mathcal{C}_L(t) = \{\{v_k, v_\ell\} \in E \mid k \leq t < \ell\}$, this also means that v_t has no neighbour in $\{v_1, \ldots, v_{t-1}\}$. Hence, $I_{max}(L)$ is an independent set.

We change L into a linear arrangement L' as follows. Let $v_t \in I_{max}(L)$ be arbitrary, and let v_ℓ be the neighbour of v_t in G with the smallest index in L. We move v_t directly to the right of the node v_ℓ, i. e., we define $L' := (v_1, \ldots, v_{t-1}, v_{t+1}, \ldots, v_\ell, v_t, v_{\ell+1}, \ldots, v_n)$. We also define $I_{max}(L')$ and $I^1_{max}(L')$ analogously as for L. By this definition, v_t has position ℓ and v_ℓ has position $\ell - 1$ in the new linear arrangement L'.

By the choice of ℓ, the cut of v_ℓ with respect to L is the same as the cut of v_t with respect to L', i. e., $\mathcal{C}_{L'}(\ell) = \mathcal{C}_L(\ell)$. Among all other vertices, only v_ℓ can have a larger cut in L' compared to L, since all other cuts either stay the same (the ones strictly to the right of v_t in L'), or do not contain the edges of v_t anymore (the ones strictly to the left of v_ℓ in L'). The only difference between the cuts $\mathcal{C}_L(\ell)$ and $\mathcal{C}_{L'}(\ell-1)$ are edges involving v_t. More precisely, all edges $\{v_t, u\}$ with $u \in N(v_t) \setminus \{v_\ell\}$ are in $\mathcal{C}_L(\ell)$ (since the vertices $N(v_t) \setminus \{v_\ell\}$ are to the right of v_ℓ in L), while $\{v_t, v_\ell\}$ is not in $\mathcal{C}_L(\ell)$ (since v_t is to the left of v_ℓ in L). In L', we have the opposite situation, i. e., none of the edges $\{v_t, u\}$ with $u \in N(v_t) \setminus \{v_\ell\}$ is in $\mathcal{C}_{L'}(\ell-1)$, while $\{v_t, v_\ell\}$ is in $\mathcal{C}_{L'}(\ell-1)$.

Summarizing, we see that all first or second order cuts for vertices $V \setminus \{v_t, v_\ell\}$ can only decrease from L to L'. In particular – which we will denote as property (†) for future reference – we observe that $\mathcal{C}_{L'}(\ell) = \mathcal{C}_L(\ell)$, and $\mathcal{C}_{L'}(\ell-1) = \mathcal{C}_L(\ell) \cup \{v_t, v_\ell\} \setminus \{\{v_t, v_u \mid u \in N(v_t) \setminus \{v_\ell\}\}$.

Towards proving $\mathsf{cw}_2(L') = \mathsf{cw}_2(G)$ we only have to check the second order cuts of v_ℓ and v_t. Since $\{v_t, v_\ell\} \in \Gamma_{L,\ell}$, (†) implies $\Gamma_{L',\ell-1} \subseteq \Gamma_{L,\ell}$. Further, $\Gamma_{L',\ell} = \mathcal{C}_{L'}(\ell-1) \cup \mathcal{C}_{L'}(\ell) = \mathcal{C}_L(\ell) \cup \{\{v_t, v_\ell\}\} \subseteq \Gamma_{L,\ell}$. Thus $\mathsf{cw}_2(L) = \mathsf{cw}_2(G)$ shows the claimed second order width of L'.

For the progress of reducing $I_{max}(L')$, first note that $v_\ell \notin I_{max}(L)$; recall that $v_t \in I_{max}(L)$ and $I_{max}(L)$ is an independent set. This implies that $|\mathcal{C}_{L'}(\ell)| = |\mathcal{C}_L(\ell)| \leq \mathsf{cw}_2(G) - 1$, so $v_t \notin I_{max}(L')$. Further, for all vertices except v_ℓ, the cut values did not increase from L to L', so $I_{max}(L') \subseteq (I_{max}(L) \setminus \{v_t\}) \cup \{v_\ell\}$ and $I^1_{max}(L') \subseteq (I^1_{max}(L) \setminus \{v_t\}) \cup \{v_\ell\}$.

We now consider two cases depending on whether or not $|\mathcal{C}_{L'}(\ell-1)|$ is strictly smaller than $|\mathcal{C}_L(\ell)| + 1$.

Case 1, $|\mathcal{C}_{L'}(\ell-1)| < |\mathcal{C}_L(\ell)| + 1$: Since $|\mathcal{C}_L(\ell)| \leq \mathsf{cw}_2(G) - 1$ (as observed above), this means that $|\mathcal{C}_{L'}(\ell-1)| < \mathsf{cw}_2(G)$. Hence, $v_\ell \notin I_{max}(L')$, which implies that $|I_{max}(L')| \leq |I_{max}(L)| - 1$.

Case 2, $|\mathcal{C}_{L'}(\ell-1)| = |\mathcal{C}_L(\ell)| + 1$: By (†) this is only possible if $N(v_t) \setminus \{v_\ell\} = \emptyset$, which means that $v_t \in I^1_{max}(L)$. Since G is a connected graph with at least three vertices, v_ℓ has at least one neighbour other than v_t, which means that $v_\ell \notin I^1_{max}(L')$. Consequently, $|I^1_{max}(L')| \leq |I^1_{max}(L)| - 1$.

Thus, we have created a linear arrangement L' with $\mathsf{cw}_2(L') = \mathsf{cw}_2(G)$ such that either $|I_{max}(L')| < |I_{max}(L)|$ or $|I_{max}(L')| = |I_{max}(L)|$ and $|I^1_{max}(L')| < |I^1_{max}(L)|$. Iterative application of this rearrangement converges to the desired linear arrangement L' with $\mathsf{cw}_2(L') = \mathsf{cw}_2(G)$ and $I_{max}(L') = \emptyset$.

Computing $\mathcal{C}_L(i)$ for each $1 \leq i \leq n$ and with this also the set $I_{max}(L)$ can be done in $\mathrm{O}(|E|)$. With this information, we only have to check the neighbourhood of the vertex that is moved for each rearrangement. Further, we only move vertices in I_{max}, so each vertex is moved at most once. Thus, all rearrangements needed to create L' from L can be done in $\mathrm{O}(|E|)$. □

Armed with this relationship, we now give the direct reduction from problem MinCutwidth to problem MinPathwidth that will in fact satisfy $\mathsf{cw}_2(G) - 1 \leq \mathsf{pw}(G') \leq \mathsf{cw}_2(G)$, effectively linking cutwidth and pathwidth with the help of Lemma 6.1. This reduction is surprisingly simple. Let $G = (V, E)$ be a simple graph.[3] We translate every original node $u \in V$ into a clique $\mathcal{K}(u) = \{u_v \mid v \in N(u)\}$, and we translate every original edge $\{u, v\}$ into an edge $\{u_v, v_u\}$. Hence, we replace each vertex u by a clique of size $|N(u)|$, and these cliques are connected according to the original graph. More formally, G is transformed into the graph $G' = (V', E')$ with $V' = \bigcup_{u \in V} \mathcal{K}(u)$ and $E' = \{\{u_v, v_u\} \mid \{u, v\} \in E\} \cup \{\{v_u, v_w\} \mid u, w \in N(v), u \neq w\}$. See Figure 8 for an illustration.

We proceed by showing the second inequality for our goal $\mathsf{cw}_2(G) - 1 \leq \mathsf{pw}(G') \leq \mathsf{cw}_2(G)$.

Lemma 6.2. *Let G be a graph with at least one edge, then $\mathsf{pw}(G') \leq \mathsf{cw}_2(G)$.*

Proof. Consider a graph $G = (V, E)$ and let $L = (v_1, \ldots, v_n)$ be an optimal linear arrangement for the second order cutwidth of G. Every node u_w of G' is either a *left node* or a *right node* according to whether u occurs to the left or to the right of w in the linear arrangement L. More formally,

[3] We discuss the case of multi-graphs later on.

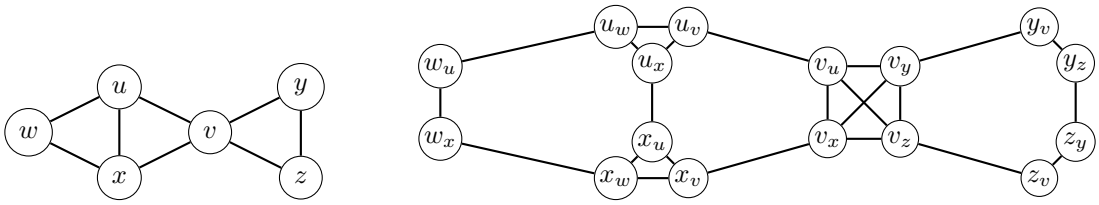

Figure 8: A graph G (left side) and the corresponding graph G' obtained by the reduction (right side). Note that vertex v of degree 4 becomes a clique $\{v_u, v_y, v_x, v_z\}$, where v_u is connected to u_v (which is a vertex of the 3-clique representing degree-3 vertex u), v_y is connected to y_v and so on.

a vertex u_w of G' is a *left node*, if $u = v_j$ and $w = v_\ell$ with $j < \ell$; the term *right node* is defined analogously. Obviously, u_w is a right node if and only if w_u is a left node.

To prove the claimed bound on the pathwidth of G', we construct a path decomposition for G' of width at most $\mathtt{cw}_2(G)$ in the form of a pd-marking scheme.

Intuitively, the pd-marking scheme is as follows. Let us first recall that, for every $i \in \{1, 2, \ldots, n\}$, $\Gamma_{L,i} = \{\{v_k, v_\ell\} \in E \mid k \leq i \leq \ell\}$, and that $\mathtt{cw}_2(G)$ is the maximum over the cardinalities of these sets. For every $i = 1, 2, \ldots, n$, we produce a step i of the marking scheme, where every edge $\{u, v\} \in \Gamma_{L,i}$ is represented by having the right node of $\{u_v, v_u\} \in E'$ set to `active` (and these are the only `active` vertices) and the left node set to `closed`. Moreover, for all edges $\{u, v\} \in E$ such that $u, v \in \{v_1, v_2, \ldots, v_{i-1}\}$, both u_v and v_u are `closed`, and all other vertices are `open`. This means that the number of `active` vertices corresponds to $|\Gamma_{L,i}|$. In order to conclude the prove, it suffices to show that (1) we can obtain step $i+1$ from step i with at most $|\Gamma_{L,i+1}| + 1$ vertices being `active` at the same time, and (2) that the cover property is satisfied. Both claims follow from how we obtain step $i+1$ from i: Let $v := v_{i+1}$. We first set all `active` right nodes from $\mathcal{K}(v_{i-1})$ to `closed`, then we set all `open` *left* nodes $v_u \in \mathcal{K}(v)$ to `active` (now all vertices from $\mathcal{K}(v)$ are `active` at the same time as required by the cover property), then for every such left node $v_u \in \mathcal{K}(V)$, we set the right node u_v to `active` and then v_u to `closed` (this is done one by one, to get at most one $|\Gamma_{L,i+1}| + 1$ `active` vertices). Now we have reached step $i+1$. Let us now define the pd-marking scheme more formally and prove its correctness.

For every $v = v_1, v_2, \ldots, v_n$, we perform the following steps.

- Step $1(v)$: Set all `open` left nodes from $\mathcal{K}(v)$ to `active`.
- Step $2(v)$: For every left node $v_u \in \mathcal{K}(v)$, set the right node $u_v \in \mathcal{K}(u)$ from `open` to `active`, and then set v_u from `active` to `closed`.
- Step $3(v)$: Set all `active` right nodes from $\mathcal{K}(v)$ to `closed`.

Note that in our intuitive explanation above, Step $3(v)$ is the first operation that we have to do in order to get from step i to step $i+1$. More precisely, after finishing Step $2(v)$ we have reached the situation that has been called step i above. It is simpler to state the Steps $1(v)$, $2(v)$ and $3(v)$ in this way, since these are exactly the operations that are with respect to the clique $\mathcal{K}(v_i)$.

By induction, it can be easily seen that after Step $1(v)$ all vertices from $\mathcal{K}(v)$ are `active` (i. e., before Step $1(v)$, only the left nodes are still `open`), after Step $2(v)$ all right nodes from $\mathcal{K}(v)$ are still `active`, but all left nodes from $\mathcal{K}(v)$ are `closed`, and after Step $3(v)$ all vertices of $\mathcal{K}(v)$ are `closed`.

Let us now prove that the marking scheme described above is a valid pd-marking scheme. Our considerations already show that we set every vertex $v \in V'$ from `open` to `active` and then from `active` to `closed`. Now let $\{p_q, r_s\}$ be an arbitrary edge of G'. If $p_q, r_s \in \mathcal{K}(v)$ for some $v \in V$, then both p_q and r_s are `active` after Step $1(v)$. If there is no $v \in V$ with $p_q, r_s \in \mathcal{K}(v)$, then, by definition of G', $p_q = u_w$ and $r_s = w_u$ with $\{u, w\} \in E$. Let us assume that u_w is a left node, which means that w_u is a right node. After Step $1(u)$, the vertex u_w is `active`. Then, in Step $2(u)$, we set $w_u \in \mathcal{K}(w)$ to `active`, before setting u_w to `closed`. Thus, both u_w and w_u are `active` at the same time. The case where w_u is a left node and u_w is a right node can be handled analogously. Consequently, the above defined marking scheme is a valid pd-marking scheme, i. e., it describes a valid path decomposition of G'.

It remains to estimate the width of the path decomposition, i. e., the maximal number of vertices that are active at the same time. We formulate the invariant: for every $i \in \{1, 2, \ldots, n\}$, as soon as Step $2(v_i)$ is done, every `active` vertex u_w is a right node such that $\{u, w\} \in \Gamma_{L,i}$.

First, we note that the invariant holds after Step $2(v_1)$, since then the set of `active` vertices is $\{w_{v_1} \mid w \in N(v_1)\}$ (which are all right vertices) and $\Gamma_{L,1} = \{\{v_1, w\} \mid w \in N(v_1)\}$. Let us now assume that the invariant holds for some $i \in \{1, 2, \ldots, n-1\}$.

Let u_w be a vertex that is `active` after Step $2(v_{i+1})$. If u_w was already `active` immediately after Step $2(v_i)$, then u_w is a right node and $\{u, w\} \in \Gamma_{L,i}$. Moreover, $u_w \notin \mathcal{K}(v_i)$, since then it would have been set to `closed` in Step $3(v_i)$. This means that $\{u, w\} \in \Gamma_{L,i+1}$. If, on the other hand, u_w was not already `active` immediately after Step $2(v_i)$, then it has been set to `active` in Step $2(v_{i+1})$ (note that all vertices set to `active` in Step $1(v_{i+1})$ are set to `closed` in Step $2(v_{i+1})$). This means that it is a right node and that $\{u, w\}$ is in $\Gamma_{L,i+1}$. Hence, as soon as Step $2(v_{i+1})$ is done, every `active` vertex u_w is a right node such that $\{u, w\} \in \Gamma_{L,i+1}$.

By induction, this proves the invariant.

Let us now estimate the maximum number of `active` vertices in the entire pd-marking scheme. For every $i \in \{1, 2, \ldots, n\}$, the number of `active` vertices after Step $2(v_i)$ is bounded by $|\Gamma_{L,i}|$ (due to the invariant). It can be easily seen that carrying out Steps $3(v_i)$, $1(v_{i+1})$ and $2(v_{i+1})$ produces a maximum of $|\Gamma_{L,i+1}| + 1$ `active` vertices. More precisely, after setting some `active` right nodes from $\mathcal{K}(v_i)$ to `closed` in Step $3(v_i)$, we set a number p of left nodes to `active` in Step $1(v_{i+1})$, which are then all set to `closed` in Step $2(v_{i+1})$, and instead we set p right nodes to `active` (which then each account for one of the `active` vertices immediately after Step $2(v_{i+1})$). However, a left node u_w is set to `closed` *immediately* after the corresponding right node w_u is set to `active`; thus, we only need one additional `active` vertex, i. e., both u_w and w_u are `active` at the same time. Consequently, the maximum number of `active` vertices of the entire pd-marking scheme is $\max\{|\Gamma_{L,i}| \mid 1 \leq i \leq n\} + 1$, which means that its width equals $\mathsf{cw}_2(L)$. □

We now give the second part of the relationship between pathwidth and the second order cutwidth. Note that we also state this result constructively, to later use it for transferring approximations.

Lemma 6.3. *Let $G = (V, E)$ be a graph with at least one edge, then $\mathsf{pw}(G') \geq \mathsf{cw}_2(G) - 1$. Further, given a path decomposition Q for G', a linear arrangement L for G with $\mathsf{cw}_2(L) \leq \mathsf{w}(Q) + 1$ can be constructed in $\mathrm{O}(|Q|)$.*

Proof. Let Q be a path decomposition for G', which we consider in the form of a pd-marking scheme with p steps. For every $v \in V$, let $\phi(v) \in \{1, 2, \ldots, p\}$ be minimal such that all vertices from $\mathcal{K}(v)$ are `active` at step $\phi(v)$ (since $\mathcal{K}(v)$ is a clique, such a $\phi(v)$ must exist). Note that $\phi(v) \neq \phi(v')$ for every $v, v' \in V$ with $v \neq v'$ (since from one step to the next at most one vertex is changed). Let $L = (v_1, v_2, \ldots, v_n)$ be the linear arrangement of G induced by the indices $\phi(v)$, i. e., for every $i, j \in \{1, 2, \ldots, n\}$, $\phi(v_i) < \phi(v_j)$ if and only if $i < j$. We note that L can be created from Q in time $\mathrm{O}(|Q|)$.

We will show that $\mathsf{w}(Q) \geq \mathsf{cw}_2(L) - 1$ (since Q is an arbitrary path decomposition, this proves the statement of the lemma). To this end, we will show that for every $i \in \{1, \ldots, n\}$ and every edge $\{u, w\} \in \Gamma_{L,i} = \{\{v_k, v_\ell\} \in E \mid k \leq i \leq \ell\}$, the vertex u_w or w_u is `active` at step $\phi(v_i)$ of Q. Since the number of `active` vertices at any step of Q is bounded by $\mathsf{w}(Q) + 1$, this implies that $|\Gamma_{L,i}| \leq \mathsf{w}(Q) + 1$, which directly implies that $\mathsf{cw}_2(L) \leq \mathsf{w}(Q) + 1$.

Let $i \in \{1, \ldots, n\}$ and let $\{u, w\} \in \Gamma_{L,i}$. For every $x \in V'$, let $I_x \subseteq \{1, 2, \ldots, p\}$ be the set of all steps of Q in which x is `active`. By definition, the sets I_x are intervals over $\{1, 2, \ldots, p\}$, and we know that $\phi(u) < \phi(v_i) < \phi(w)$. Since I_{u_w} contains $\phi(u)$, I_{w_u} contains $\phi(w)$, and $I_{u_w} \cap I_{w_u} \neq \emptyset$ (since there has to be a step where both u_w and w_u are `active`), we conclude that $\{\phi(u), \phi(u) + 1, \ldots, \phi(w)\} \subseteq I_{u_w} \cup I_{w_u}$. Thus, we also have that $\phi(v_i) \in I_{u_w} \cup I_{w_u}$. This means that u_w or w_u is `active` at step $\phi(v_i)$ of Q. □

As we want to transfer approximations for MinPathwidth to MinCutwidth also for multigraphs, we briefly explain how all results of this section easily generalize to this setting. The relationship of second order cutwidth and cutwidth remains exactly the same; the proof of Lemma 6.1 generalizes to multigraphs with the only adjustment that $I^1_{max}(L)$ is defined as the set of vertices

in $I_{max}(L)$ that have exactly one neighbour (that can be connected by multiple edges, so in this sense not of degree one).

For the connection to pathwidth, we can extend the reduction described before Lemma 6.2 to multigraphs in a straightforward way. Let G be a multigraph and let $\{u, v\}$ be an edge of G with some multiplicity t (i. e., in G there are t parallel edges going from u to v). While in the simple graph case an edge $\{u, v\}$ of G was translated into the single edge $\{u_v, v_u\}$ of G', we will now use t simple edges $\{u_v^i, v_u^i\}$, $1 \leq i \leq t$, in order to represent the multiplicity t of the multi-edge of G. Hence, we represent a single vertex v of G with $N(v) = \{u_1, u_2, \ldots, u_k\}$ by several vertices $\mathcal{K}(v) = \{v_{u_1}^1, \ldots, v_{u_1}^{t_1}, v_{u_2}^1, \ldots, v_{u_2}^{t_2}, \ldots, v_{u_k}^1, \ldots, v_{u_k}^{t_k}\}$, where the $t_1, t_2, \ldots, t_k$ are the multiplicities of the edges between v and its neighbours $u_1, u_2, \ldots, u_k$. Analogously to the simple graph case, we connect all the vertices of $\mathcal{K}(v)$ into a clique.

We can now prove Lemma 6.2 for the case of multi-graphs in a similar way as for simple graphs. For a given multi-graph G, we apply the reduction from above, which yields a *simple* graph G' (recall that the multiplicities are represented by individual vertices in the cliques $\mathcal{K}(v)$ with $v \in V$). Then, we fix again an optimal linear arrangement $L = (v_1, \ldots, v_n)$ for the second order cutwidth of G. We call a node u_w^i a *left node* if u occurs to the left of w with respect to L, and *right nodes* are defined analogously. Now, we can define a pd-marking scheme in the same way as in the proof of Lemma 6.2, i. e., for every $v = v_1, v_2, \ldots, v_n$, we perform the Steps $1(v)$, $2(v)$ and $3(v)$. Since also in the adapted reduction we have the cliques $\mathcal{K}(v)$, both Step $1(v)$ (i. e., set all `open` left nodes from $\mathcal{K}(v)$ to `active`) and Step $3(v)$ (i. e., set all `active` right nodes from $\mathcal{K}(v)$ to `closed`) apply verbatim in the same way, while Step $2(v)$ reads as follows: For every left node $v_u^i \in \mathcal{K}(v)$, set the right node $u_v^i \in \mathcal{K}(u)$ from `open` to `active`, and then set v_u^i from `active` to `closed`. The proof that this gives a path decomposition of width at most $\mathsf{cw}_2(G)$ then is completely analogous.

The same holds for the proof of Lemma 6.3.

We are now ready to state the main result of this section, i. e., how pathwidth approximation carries over to cutwidth approximation.

Lemma 6.4. *If there is an $r(\mathsf{opt}, |V|)$-approximation algorithm for* MinPathwidth *with running-time $\mathrm{O}(f(|V|))$, then there is also an $2r(2\,\mathsf{opt}, h)$-approximation algorithm for* MinCutwidth *on multigraphs with running time $\mathrm{O}(f(h) + h^2 + n)$, where n is the number of vertices and h is the number of edges.*

Proof. Let $G = (V, E)$ be an instance of MinCutwidth and let $\mathcal{A}$ be an $r(\mathsf{pw}(G'), |V|)$-approximation for MinPathwidth Lemma 6.2 combined with Lemma 6.1 shows that $\mathsf{pw}(G') \leq \mathsf{cw}_2(G) \leq 2\,\mathsf{cw}(G)$. Further, Lemma 6.3 shows that any path-decomposition P of width k for G' can be translated into a linear arrangement L for G with $\mathsf{cw}_2(L) \leq k+1$ in $\mathrm{O}(|P|)$. By Lemma 6.1, we can compute then from L a linear arrangement L' with $\mathsf{cw}(L') \leq \mathsf{cw}_2(L) - 1 \leq k$ in $\mathrm{O}(h)$.

The relative error of L' can thus be bounded by $R(G, L) = \frac{\mathsf{cw}(L')}{\mathsf{cw}(G)} \leq \frac{2\,\mathsf{pw}(P)}{\mathsf{pw}(G')} = 2R(G', P)$. The algorithm which builds G' from G in $\mathrm{O}(n + h)$, runs $\mathcal{A}$ on G' in $\mathrm{O}(f(h))$ and creates a linear arrangement L' in $\mathrm{O}(h + |P|)$ has a performance ratio $2r(\mathsf{pw}(G'), |V|) \leq 2r(2\,\mathsf{cw}(G), h)$ and an overall running time in $\mathrm{O}(f(h) + h)$ (note that $\mathrm{O}(|P|) \subseteq \mathrm{O}(f(h))$, since $\mathcal{A}$ builds P in $\mathrm{O}(f(|V(G')|)) = \mathrm{O}(f(h))$). □

For example, if we apply this lemma with respect to the $\mathrm{O}(\log n\sqrt{\log \mathit{opt}})$-approximation algorithm of [21], we obtain an $\mathrm{O}(\sqrt{\log(\mathsf{opt})}\log(h))$-approximation algorithm for MinCutwidth on multigraphs with h edges, and if we apply it with respect to the $\mathrm{O}(\mathsf{tw}\sqrt{\log \mathsf{tw}})$-approximation algorithm of [30], we obtain an $\mathrm{O}(\sqrt{\log(\mathsf{opt})}\,\mathsf{opt})$-approximation algorithm. Note that the second result holds since an $\mathrm{O}(\mathsf{tw}\sqrt{\log \mathsf{tw}})$-approximation algorithm for pathwidth is also an $\mathrm{O}(\mathsf{opt}\,\sqrt{\log \mathsf{opt}})$-approximation algorithm for pathwidth. Unfortunately, our reduction blows up the treewidth, so it does not give a translation to a ratio that depends only on the treewidth.

To the best knowledge of the authors, these are new approximations ratios for cutwidth that have not previously been reported in the literature, and that are better or incomparable to existing ones. Hence, let us state this result more prominently.

Corollary 6.5. *There is a (polynomial-time) $\mathrm{O}(\sqrt{\log(\mathsf{opt})}\log(h))$-approximation algorithm and an $\mathrm{O}(\sqrt{\log(\mathsf{opt})}\,\mathsf{opt})$-approximation algorithm for* MinCutwidth *on multigraphs with h edges.*

7 Conclusions

In this work, we have answered several open questions about the string parameter of the locality number. Our main tool was to relate the locality number to the graph parameters cutwidth and pathwidth via suitable reductions. As an additional result, our reductions also pointed out an interesting relationship between these classical graph parameters and the locality number for strings, with implications for approximating these parameters.

While our focus is on theoretical results in form of lower and upper complexity bounds, we stress here that the reductions may also be of practical interest, since they allow to transform any practical pathwidth or cutwidth algorithm into a practical algorithm for computing the locality number (or to transform a practical pathwidth algorithm into a practical algorithm for computing the cutwidth). This seems particularly interesting, since, as pointed out at the end of Section 5.2, practical algorithms for constructing path decompositions of small width is a vibrant research area of practical algorithm engineering.

References

[1] Amihood Amir and Igor Nor. Generalized function matching. *Journal of Discrete Algorithms*, 5(3):514–523, 2007. doi:10.1016/j.jda.2006.10.001.

[2] Dana Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21(1):46–62, 1980. doi:10.1016/0022-0000(80)90041-0.

[3] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. *Journal of the ACM*, 62(5):42:1–42:25, 2015. doi:10.1145/2775105.

[4] Sanjeev Arora, Satish Rao, and Umesh V. Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM*, 56(2):5:1–5:37, 2009. doi:10.1145/1502793.1502794.

[5] Giorgio Ausiello, Alberto Marchetti-Spaccamela, Pierluigi Crescenzi, Giorgio Gambosi, Marco Protasi, and Viggo Kann. *Complexity and approximation: combinatorial optimization problems and their approximability properties*. Springer, 1999. doi:10.1007/978-3-642-58412-1.

[6] Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In Rafail Ostrovsky, editor, *52nd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2011*, pages 472–481. IEEE, 2011. doi:10.1109/focs.2011.95.

[7] Pablo Barceló, Leonid Libkin, Anthony W. Lin, and Peter T. Wood. Expressive languages for path queries over graph-structured data. *ACM Transactions on Database Systems*, 37(4):1–46, 2012. doi:10.1145/2389241.2389250.

[8] Hans L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11(1–2):1–21, 1993. URL: http://www.inf.u-szeged.hu/actacybernetica/edb/vol11n1_2/pdf/Bodlaender_1993_ActaCybernetica.pdf.

[9] Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(5):1305–1317, 1996. doi:10.1137/s0097539793251219.

[10] Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1–2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

[11] Hans L. Bodlaender. Fixed-parameter tractability of treewidth and pathwidth. In Hans L. Bodlaender, Rod Downey, Fedor V. Fomin, and Dániel Marx, editors, *The Multivariate Algorithmic Revolution and Beyond*, volume 7370 of *LNCS*, pages 196–227, 2012. doi:10.1007/978-3-642-30891-8_12.

[12] Hans L. Bodlaender, Fedor V. Fomin, Arie M. C. A. Koster, Dieter Kratsch, and Dimitrios M. Thilikos. A note on exact algorithms for vertex ordering problems on graphs. *Theory of Computing Systems*, 50(3):420–432, 2012. doi:10.1007/s00224-011-9312-0.

[13] Katrin Casel, Joel D. Day, Pamela Fleischmann, Tomasz Kociumaka, Florin Manea, and Markus L. Schmid. Graph and string parameters: Connections between pathwidth, cutwidth and the locality number. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, pages 109:1–109:16, 2019. doi:10.4230/LIPIcs.ICALP.2019.109.

[14] David Coudert, Dorian Mazauric, and Nicolas Nisse. Experimental evaluation of a branch-and-bound algorithm for computing pathwidth and directed pathwidth. *ACM Journal of Experimental Algorithmics*, 21(1):1.3:1–1.3:23, 2016. doi:10.1145/2851494.

[15] Joel D. Day, Pamela Fleischmann, Florin Manea, and Dirk Nowotka. Local patterns. In Satya V. Lokam and R. Ramanujam, editors, *Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017*, volume 93 of *LIPIcs*, pages 24:1–24:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.FSTTCS.2017.24.

[16] Joel D. Day, Pamela Fleischmann, Florin Manea, Dirk Nowotka, and Markus L. Schmid. On matching generalised repetitive patterns. In Mizuho Hoshi and Shinnosuke Seki, editors, *Developments in Language Theory, DLT 2018*, volume 11088 of *LNCS*, pages 269–281. Springer, 2018. doi:10.1007/978-3-319-98654-8_22.

[17] Holger Dell, Christian Komusiewicz, Nimrod Talmon, and Mathias Weller. The PACE 2017 parameterized algorithms and computational experiments challenge: The second iteration. In Daniel Lokshtanov and Naomi Nishimura, editors, *Parameterized and Exact Computation, IPEC 2017*, volume 89 of *LIPIcs*, pages 30:1–30:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.IPEC.2017.30.

[18] Josep Díaz, Jordi Petit, and Maria Serna. A survey of graph layout problems. *ACM Computing Surveys*, 34(3):313–356, 2002. doi:10.1145/568522.568523.

[19] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

[20] Thomas Erlebach, Peter Rossmanith, Hans Stadtherr, Angelika Steger, and Thomas Zeugmann. Learning one-variable pattern languages very efficiently on average, in parallel, and by asking queries. *Theoretical Computer Science*, 261(1):119–156, 2001. doi:10.1016/s0304-3975(00)00136-5.

[21] Uriel Feige, MohammadTaghi HajiAghayi, and James R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM Journal on Computing*, 38(2):629–657, 2008. doi:10.1137/05064299x.

[22] Henning Fernau, Florin Manea, Robert Mercas, and Markus L. Schmid. Pattern matching with variables: Fast algorithms and new hardness results. In Ernst W. Mayr and Nicolas Ollinger, editors, *Symposium on Theoretical Aspects of Computer Science, STACS 2015*, volume 30 of *LIPIcs*, pages 302–315. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.STACS.2015.302.

[23] Henning Fernau, Florin Manea, Robert Mercaş, and Markus L. Schmid. Revisiting Shinohara's algorithm for computing descriptive patterns. *Theoretical Computer Science*, 733:44–54, 2018. doi:10.1016/j.tcs.2018.04.035.

[24] Henning Fernau and Markus L. Schmid. Pattern matching with variables: A multivariate complexity analysis. *Information and Computation*, 242:287–305, 2015. doi:10.1016/j.ic.2015.03.006.

[25] Henning Fernau, Markus L. Schmid, and Yngve Villanger. On the parameterised complexity of string morphism problems. *Theory of Computing Systems*, 59(1):24–51, 2016. doi:10.1007/s00224-015-9635-3.

[26] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer, 2006. doi:10.1007/3-540-29953-X.

[27] Dominik D. Freydenberger. Extended regular expressions: Succinctness and decidability. *Theory of Computing Systems*, 53(2):159–193, 2013. doi:10.1007/s00224-012-9389-0.

[28] Dominik D. Freydenberger and Markus L. Schmid. Deterministic regular expressions with back-references. *Journal of Computer and System Sciences*, 2019. doi:10.1016/j.jcss.2019.04.001.

[29] Jeffrey E. F. Friedl. *Mastering Regular Expressions*. O'Reilly, Sebastopol, CA, 3rd edition, 2006.

[30] Carla Groenland, Gwenaël Joret, Wojciech Nadara, and Bartosz Walczak. Approximating pathwidth for graphs of small treewidth. *ACM Trans. Algorithms*, 19(2):16:1–16:19, 2023. doi:10.1145/3576044.

[31] Venkatesan Guruswami and Ali Kemal Sinop. Lasserre hierarchy, higher eigenvalues, and approximation schemes for graph partitioning and quadratic integer programming with PSD objectives. In Rafail Ostrovsky, editor, *52nd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2011*, pages 482–491. IEEE, 2011. doi:10.1109/FOCS.2011.36.

[32] Carl Hierholzer and Christian Wiener. Über die Möglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechung zu umfahren. *Mathematische Annalen*, 6(1):30–32, 1873. doi:10.1007/bf01442866.

[33] Juhani Karhumäki, Filippo Mignosi, and Wojciech Plandowski. The expressibility of languages and relations by word equations. *Journal of the ACM*, 47(3):483–505, 2000. doi:10.1145/337244.337255.

[34] Michael Kearns and Leonard Pitt. A polynomial-time algorithm for learning k-variable pattern languages from examples. In Ronald L. Rivest, David Haussler, and Manfred K. Warmuth, editors, *Computational Learning Theory, COLT 1989*, pages 57–71. Morgan Kaufmann, 1989. doi:10.1016/b978-0-08-094829-4.50007-6.

[35] Subhash Khot. On the power of unique 2-prover 1-round games. In John H. Reif, editor, *34th Annual ACM Symposium on Theory of Computing, STOC 2002*, pages 767–775. ACM, 2002. doi:10.1145/509907.510017.

[36] Subhash Khot. On the unique games conjecture (invited survey). In *Computational Complexity, CCC 2010*, pages 99–121. IEEE, 2010. doi:10.1109/CCC.2010.19.

[37] Ton Kloks, editor. *Treewidth, Computations and Approximations*, volume 842 of *LNCS*. Springer, 1994. doi:10.1007/BFb0045375.

[38] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46(6):787–832, 1999. doi:10.1145/331524.331526.

[39] M. Lothaire, editor. *Algebraic Combinatorics on Words*. Cambridge University Press, 2002. doi:10.1017/cbo9781107326019.

[40] Fillia Makedon, Christos H. Papadimitriou, and Ivan Hal Sudborough. Topological bandwidth. *SIAM Journal on Algebraic and Discrete Methods*, 6(3):418–444, 1985. doi:10.1137/0606044.

[41] Florin Manea and Markus L. Schmid. Matching patterns with variables. In *Combinatorics on Words - 12th International Conference, WORDS 2019, Loughborough, UK, September 9-13, 2019, Proceedings*, pages 1–27, 2019. doi:10.1007/978-3-030-28796-2_1.

[42] Yen Kaow Ng and Takeshi Shinohara. Developments from enquiries into the learnability of the pattern languages from positive data. *Theoretical Computer Science*, 397(1–3):150–165, 2008. doi:10.1016/j.tcs.2008.02.028.

[43] Jordi Petit. Addenda to the survey of layout problems. *Bulletin of the EATCS*, 105:177–201, 2011. URL: http://eatcs.org/beatcs/index.php/beatcs/article/view/98.

[44] Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In Leonard J. Schulman, editor, *42nd ACM Symposium on Theory of Computing, STOC 2010*, pages 755–764. ACM, 2010. doi:10.1145/1806689.1806792.

[45] Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *Computational Complexity, CCC 2012*, pages 64–73. IEEE, 2012. doi:10.1109/CCC.2012.43.

[46] Daniel Reidenbach. Discontinuities in pattern inference. *Theoretical Computer Science*, 397(1–3):166–193, 2008. doi:10.1016/j.tcs.2008.02.029.

[47] Daniel Reidenbach and Markus L. Schmid. Patterns with bounded treewidth. *Information and Computation*, 239:87–99, 2014. doi:10.1016/j.ic.2014.08.010.

[48] Markus L. Schmid. Characterising REGEX languages by regular languages equipped with factor-referencing. *Information and Computation*, 249:1–17, 2016. doi:10.1016/j.ic.2016.02.003.

[49] Takeshi Shinohara. Polynomial time inference of pattern languages and its application. In *7th IBM Symposium on Mathematical Foundations of Computer Science*, pages 191–209, 1982.

[50] Karol Suchan and Yngve Villanger. Computing pathwidth faster than 2^n. In Jianer Chen and Fedor V. Fomin, editors, *Parameterized and Exact Computation, IWPEC 2009*, volume 5917 of *LNCS*, pages 324–335. Springer, 2009. doi:10.1007/978-3-642-11269-0_27.

[51] Dimitrios M. Thilikos, Maria J. Serna, and Hans L. Bodlaender. Cutwidth I: A linear time fixed parameter algorithm. *Journal of Algorithms*, 56(1):1–24, 2005. doi:10.1016/j.jalgor.2004.12.001.

[52] Yu (Ledell) Wu, Per Austrin, Toniann Pitassi, and David Liu. Inapproximability of treewidth and related problems. *J. Artif. Intell. Res.*, 49:569–600, 2014. URL: https://doi.org/10.1613/jair.4030, doi:10.1613/JAIR.4030.

A Additional Word-Combinatorial Considerations

In this section, we give the details that have been omitted in Section 3.

A.1 The Locality of the Zimin Words

Lemma A.1. *$\mathsf{loc}(Z_i) = \frac{|Z_i|+1}{4} = 2^{i-2}$ for $i \in \mathbb{N}_{\geq 2}$.*

Proof. Clearly, x_1 and $x_1x_2x_1$ are 1-local. Consider a fixed $i \in \mathbb{N}$ and the marking sequence $(x_2, x_1, y_1, y_2, \ldots, y_{i-2})$ for $i \geq 3$ and $\{y_1, \ldots, y_{i-2}\} = \{x_3, \ldots, x_i\}$. Notice that for all $j \in \mathbb{N}$, x_j occurs 2^{i-j} times in Z_i. Thus by marking x_2, there are 2^{i-2} marked blocks. Since all occurrences of x_1 are adjacent to occurrences of x_2, marking x_1 does not change the number of marked blocks. As marking the remaining variables only leads to the merging of some pairs of consecutive blocks into one, we never have more than 2^{i-2} marked blocks.

In the following we will show the converse. More precisely, we show that if a sequence is optimal for Z_i then it starts with x_2, x_1. Let us note first that, for $2 \leq p < r$, between two consecutive

occurrences of x_r in Z_i there is one occurrence of x_p. More precisely, each occurrence of a variable x_p, with $p \geq 2$, is directly between two occurrences of x_1. Also, notice that x_j has 2^{i-j} occurrences in Z_i. Now, if x_1 is marked before x_2, because Z_i starts with x_1x_2 and ends with x_2x_1, it is immediate that after the marking of x_1 we will have at least $2^{i-2}+1$ marked blocks in the word (separated by the 2^{i-2} unmarked occurrences of x_2). This is, thus, a marking sequence that is not optimal. So x_2 is marked before x_1 in an optimal sequence. Assume that there exists x_j, with $j > 2$, which is also marked before x_1 in an optimal sequence. Let w be a word such that $Z_i = x_1wx_1$. There are $2^{i-1}-2$ occurrences of x_1 in w, and w starts with x_2x_1 and ends with x_1x_2. As each two consecutive (marked) occurrences of the letters x_2 and x_j are separated by unmarked occurrences of x_1 we have that, just before marking x_1, there are at least $\min\{2^{i-1}-1, 2^{i-2}+2^{i-j}\}$ marked blocks in w (and the same number in Z_i). This again shows that this is not an optimal marking sequence. So, before x_1 is marked, only x_2 should be marked. This concludes the proof of our claim, and of the proposition. □

A.2 The Locality of (Condensed) Palindromes and Repetitions.

We use the following notation. Given a marking sequence σ, let σ^R be the marking sequence obtained by reversing σ (i.e. $\sigma^R(i) = \sigma(|X| - i + 1)$ for $1 \leq i \leq |X|$).

By $\mathsf{loc}(\mathsf{cond}(w)) = \mathsf{loc}(w)$, it is enough to show our results for condensed words. Since there are no condensed palindromes of even length, only palindromes of odd length are of interest when determining the locality number. A word w is called strictly k-local if for every optimal marking sequence of w there is a stage when exactly k factors are marked. For a letter $a \in \mathsf{alph}(w)$, we denote by $|w|_a$ the number of occurrences of a in w. For simplicity of notations, let $[n] := \{1, 2, \ldots, n\}$.

Let $w_i \in (X \cup \overline{X})^*$ be the marked version of w at stage $i \in [|\,\mathsf{alph}(w)|]$ for a given marking sequence σ.

Lemma A.2. *Define the morphism $f : X \cup \overline{X} \to \{0, 1\}$ by*

$$f(x) = \begin{cases} 0 & \text{if } x \in X, \\ 1 & \text{if } x \in \overline{X}. \end{cases}$$

If w is a palindrome and σ a marking sequence for w then $f(w_i)$ is a palindrome for all $i \in [|\,\mathsf{alph}(w)|]$.

Proof. Let $w = uxu^R$ be a palindrome with $u \in X^*$ and $x \in X \cup \overline{X}$ and $|w| = n \in \mathbb{N}$. Moreover let σ be a marking sequence for w and $i \in [|\,\mathsf{alph}(w)|]$. Since w is a palindrome, $w[j] = w[n-j]$. This implies $w_i[j], w_i[n-j]$ are both either in X or in $\overline{X}$. Thus either are both mapped to 0 or to 1. Consequently $f(w_i)$ is a palindrome. □

Recall the definition of *border priority markable* from [15]. A strictly k-local word $w = avb \in XX^*X$ is called border priority markable if there exists a marking sequence σ of w such that in every stage $i \in [|\alpha(w)|]$ of σ where k blocks are marked, a and b are marked as well. Analogously right-border priority markable and left-border priority markable are defined: A strictly k-local word $w = avb \in XX^*X$ is called right-border priority markable (rbpm) if if there exists a marking sequence σ of w such that in every stage $i \in [|\alpha(w)|]$ of σ where k blocks are marked, b is marked as well - respectively, for left-border priority markable, a is marked as well.

Remark A.3. If $w \in X^*$ is right-border priority markable, then u^R is left-border priority markable.

Lemma A.4. *Let $w = u\mathtt{a}u^R$ be an odd-length condensed palindrome with $u \in X^*$ and $\mathtt{a} \in X$. Let u be strictly k-local witnessed by the marking sequence σ.*

- *If u is rbpm then $\mathsf{loc}(w) = 2k - 1$,*
- *if u is not rbpm and $\mathtt{a} \notin \mathsf{alph}(u)$ then $\mathsf{loc}(w) = 2k$,*
- *if u is not rbpm and $\mathtt{a} \in \mathsf{alph}(u)$ and for all optimal marking sequences for u there exists a stage $i \in [|\,\mathsf{alph}(u)|]$ such that $\mathtt{a}$ is marked, k blocks are marked, and $u[|u|]$ is unmarked then $\mathsf{loc}(w) = 2k + 1$, and*

- *else* $\mathsf{loc}(w) = 2k$.

Proof. Let σ be an optimal marking sequence of u. If $\mathsf{a} \in \mathsf{alph}(u)$ then σ is a marking sequence for w. Marking w w.r.t. σ leads to $\pi_\sigma(w) \leq 2k+1$ since there are at most maximal k blocks marked each in u and u^R, and additionally the single a in the middle. If $\mathsf{a} \notin \mathsf{alph}(u)$ then $\sigma' = \sigma \cup \{(|u|+1, \mathsf{a}\}$ is a marking sequence for w with $\pi_{\sigma'}(w) \leq 2k$, since by marking w.r.t. σ maximal k blocks are marked by σ each in u and u^R and afterwards on marking a two blocks are joined. Thus in any case $\mathsf{loc}(w) \leq 2k+1$.
case 1. Consider u to be rbpm. Thus in every stage $i \in [|\,\mathsf{alph}(u)|]$ where k blocks are marked, $u[|u|]$ is marked. This implies that $\pi_\sigma(w) \leq 2k-1$ or $\pi_{\sigma'}(w) \leq 2k-1$ with σ' defined as above.
Supposition: $\mathsf{loc}(w) =: \ell < 2k-1$
Let μ be an optimal marking sequence for w. Then μ is also a marking sequence for u and thus $\pi_\mu(u) \geq k$. By $\mathsf{loc}(u) = k$ there exists a stage $i \in [|\,\mathsf{alph}(w)|]$ of μ such that k blocks are marked in u, or more precisely $|\,\mathsf{cond}(f(u_i))|_1 = k$. On the other hand $|\,\mathsf{cond}(f(w_i))|_1 \leq \ell$. Since u is rbpm $u[|u|]$ is marked. If x is not marked, $|\,\mathsf{cond}(f(u_i))|_1 \leq \frac{\ell}{2} < \frac{2k-1}{2} = k - \frac{1}{2}$. If x is marked, $|\,\mathsf{cond}(f(u_i))|_1 \leq \frac{\ell-1}{2} < \frac{2k-2}{2} = k-1$. This is in both cases a contradiction to $|\,\mathsf{cond}(f(u_i))|_1 = k$.
case 2. Consider now that u is not rbpm. Thus there exists a stage $i \in [|\,\mathsf{alph}(u)|]$ in which k blocks are marked but $u[|u|]$ is unmarked. If a is not in $\mathsf{alph}(u)$ marking a before stage i leads to $2k+1$ blocks for the largest such i. Considering σ' then at the beginning u and u^R are completely marked and in the end two blocks are joined by marking a. This leads to $\mathsf{loc}(w) \leq 2k$.
Supposition: $\mathsf{loc}(w) < 2k$
As described, a needs to be marked after the last stage where in u k blocks are marked without $u[|u|]$ being marked. But this sums up to k blocks marked in u and k blocks marked in u^R, hence overall $2k$ blocks. This concludes the case $\mathsf{a} \notin \mathsf{alph}(u)$.
Consider $\mathsf{a} \in \mathsf{alph}(u)$ and assume that a is marked by σ when k blocks are marked in u and $u[|u|]$ is unmarked. Thus $\pi_\sigma(w) = 2k+1$.
Supposition: $\mathsf{loc}(w) =: \ell < 2k+1$
Let μ be an optimal marking sequence for w.
Additional supposition: μ not optimal for u
Then there exists a stage $i \in [\mathsf{alph}(w)]$ such that $|\,\mathsf{cond}(f(u_i))|_1 = k+1$. If a is unmarked in this stage, $|\,\mathsf{cond}(f(w_i))|_1 = 2k+2 > \ell$ which contradicts the first supposition. If a is marked in this stage $|\,\mathsf{cond}(f(w_i))|_1 = 2k+1$ which contradicts the first supposition.
Thus, μ is optimal for u. By assumption there exists a stage $i \in [|\,\mathsf{alph}(u)|]$ such that a is marked, k blocks are marked, and $u[|u|]$ is unmarked. This implies since $\mathsf{cond}(f(w_i))$ is a palindrome that at most $\frac{\ell-1}{2}$ blocks are marked in u. Thus, $k \leq \frac{\ell-1}{2} < \frac{2k+1-1}{2} = k$.
case 3. In the remaining case u is not rbpm, $\mathsf{a} \in \mathsf{alph}(u)$, and there exists an optimal marking sequence for u such that in every stage a is unmarked or less than k blocks are marked or $u[|u|]$ is marked. Let σ be such a marking sequence. Then $\pi_\sigma(w) = 2k$.
Supposition: $\mathsf{loc}(w) =: \ell < 2k$
Let μ be an optimal marking sequence for w. Since u is not rbpm there exists a stage $i \in [|\,\mathsf{alph}(u)|]$ such that $|\,\mathsf{cond}(f(u_i))|_1 = k$ and $u[|u|]$ is unmarked. If a were unmarked in stage i, $k = |\,\mathsf{cond}(f(u_i))|_1 \leq \frac{\ell}{2} < k$ and if a were marked in stage i, $k = |\,\mathsf{cond}(f(u_i))|_1 \leq \frac{\ell-1}{2} < \frac{2k-1}{2} = k - \frac{1}{2}$. Thus $2k+1 \leq \ell < 2k$ would hold. □

Lemma A.5. *Let $w = u^i$ be the i-times repetition for $u \in X^*$ and $i \in \mathbb{N}$. If u is strictly k-local then*

$$\mathsf{loc}(w) = \begin{cases} ik - i + 1, & \text{if } u \text{ is bpm,} \\ ik, & \text{otherwise.} \end{cases}$$

Proof. Let σ be a marking sequence with $\pi_\sigma = \mathsf{loc}(u) = k$. Since $\mathsf{alph}(u) = \mathsf{alph}(u^i)$ for all $i \in \mathbb{N}$, σ is also a marking sequence for w. If u is not bpm, there exists a stage during the marking in which k blocks are marked by σ and at least one of $u[1]$ or $u[|u|]$ is unmarked. Thus marking w according to the sequence σ leads to $\pi_\sigma(w) = ik$. If u is bpm, in any stage in which k blocks are marked, $u[1]$ and $u[|u|]$ are marked and thus in w, while being marked according to σ, the last marked block of an occurrence of u and the first marked block of the next occurrence of u coincide, as soon as the prefix of length $|u|$ of w contains k marked blocks. So, we get $\pi_\sigma(w) = ik - i + 1$.

For proving $\mathsf{loc}(w) = ik$ or $\mathsf{loc}(w) = ik - i + 1$ respectively, consider firstly $i = 2$. Assume first that w is bpm. Suppose $\mathsf{loc}(w) = \ell < 2k - 1$. Let σ' be the marking sequence witnessing $\mathsf{loc}(w) = \ell$. Since u is strictly k-local, there exists a stage in marking w by σ' in which u has k marked blocks. The second u has exactly as many marked blocks as the first one, so also k. In the best case, in w the last marked block of the first u and the first marked block of the second u are connected. Anyway, the number of marked blocks of w is, in that case, exactly $2k - 1$. A contradiction to the assumption $\mathsf{loc}(w) = \ell < 2k - 1$. If u is not bpm, then, once again, there exists a stage in marking w by σ' in which u has k marked blocks. The second u has also exactly k marked block. But, in this case, in w the last marked block of the first u and the first marked block of the second u do not touch (as either the last letter of u or its first letter are not marked). So w has $2k$ marked blocks, a contradiction.

This reasoning can be trivially extended for $i > 2$. □